Performance of the ABR policer in an ATM network

Nagaraja Srikantashastry

University of Wollongong

Follow this and additional works at: https://ro.uow.edu.au/theses

Recommended Citation
Performance of the ABR policer in an ATM network.

A thesis submitted to the
University of Wollongong
in fulfilment of the requirements for the degree of
Masters (Honours)

by

Nagaraja Srikantashastry
Bachelor of Engineering (Electronics and Telecommunications)
National Institute of Engineering
Mysore 1981

November 2000
Abstract

Both the Available Bit Rate (ABR) and Unspecified Bit Rate (UBR) services in Asynchronous Transfer Mode (ATM) networks cater for data traffic. The network guarantees low cell loss as a parameter for Quality of Service (QoS) in the ABR service, but not in UBR service. In order to achieve a low cell-loss ratio (CLR) for the ABR service, it is critical for the network to adopt suitable traffic management functions before and after the admission of a call. Before any call admission, the network reserves resources through the Call Admission Control (CAC) function, based on the agreed traffic contract. After a call admission, the network monitors, through a policing function, any violation of the source rate and behaviour agreed to in the contract. The policer checks the conformance (as per the conformance definition [1]) of incoming cells, passes those which do conform and tags/discards those which do not conform. The performance accuracy of a policer plays a key role in a network’s ability to meet the guaranteed QoS. Any errors in the policing algorithm leads to poor performance, with a potential QoS short fall. This thesis proposes a performance measurement method for the ABR policer.

In the ABR policer inaccuracies arise from two types of errors. Type 1 errors occur when the policer tags a conformed cell from a well-behaved user and Type 2 errors occur when the policer passes a non-conformed cell from a misbehaved source. A lower police rate monitoring a higher source rate (adopted based on feedback) causes Type 1 errors whereas a higher police rate monitoring a lower source rate (adopted based on feedback) causes Type 2 errors. Tagging more cells due to Type 1 errors will affect the CLR and passing excess cells due to Type 2 errors will affect the network load and bandwidth sharing. This thesis proposes a quantitative measurement of tagged and excess cells as a measure of the performance of a policer.
The need for an accurate policer to monitor the source traffic of ATM networks is a key requirement. In order to identify an accurate policer, the quantification of performance in terms of excess cells and tagged cells is needed. This requires a model of the ABR policer located at the UNI. This thesis presents such a model.

The ABR policer comprises two algorithms, one for conformance checking with reference to a police rate, the other for generating the police rate. Different algorithms are available as options to implement the ABR policer. The ATM Forum has specified a Dynamic Generic Cell Rate Algorithm (DGCRA) for cell conformance checking with reference to a police rate which is generated by Algorithm A or 2_Store Algorithm B. These police rates are based on the Explicit Rates (ERs) sent across the UNI as feedback towards the source through backward Resource Management (RM) cells. Algorithm A is designed to be the most accurate because it memorises all the feedback rates, however it is more complex to implement. The 2_Store algorithm is found to be a simple algorithm memorising few feedback rates (two at the most) and thus it is less accurate. The N_Store, an extension of the 2_Store algorithm has been proposed to the ATM Forum [3] as more accurate when compared to 2_Store policer and it is equivalent to Algorithm A. This thesis compares the performance of algorithms A, 2_Store and N_Store in terms of tagged cells and excess cells. After the analysis, the thesis concludes that Algorithm A is the most accurate policer producing least tagged and excess cells compared to 2_Store and N_Store algorithms. The thesis, by using the proposed performance measurement techniques establishes that the N_Store algorithm is not accurate and produces loss. Thus the thesis disproves the authors’ statement “the contribution recommends Algorithm A as the basis of policing ABR, and recommends a cyclic n-store algorithm as a practical realisation of Algorithm A” [3].

The thesis also investigates the cause for the inaccuracy of N_Store algorithm and finds inherent errors in the algorithm. Corrections to fix these errors are proposed. The thesis, through performance measurement, shows that N_Store algorithm after the correction is equivalent to Algorithm A.

The thesis addresses the requirements of a simple, less cell loss and accurate policer that is equivalent to Algorithm A. In order to fulfil these requirements, A simplified
N_Store_I algorithm with all the above corrections is proposed. The thesis measures the performance and shows that N_Store_I is equivalent to Algorithm A.
I would like to thank my supervisor, Dr. Tony Eyers. Without his help encouragement and support at difficult times, it would not have been possible for me to complete this work. I would like to thank my wife, Jyothi, and mother, Susheelamma, in supporting the completion of this work. I should not forget to thank my daughter, Minnie, and son, Shinku, who supported me by allowing me to do my work peacefully, without many disturbances.
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABR</td>
<td>Available Bit Rate</td>
</tr>
<tr>
<td>ACR</td>
<td>Allowed Cell Rate</td>
</tr>
<tr>
<td>ADTF</td>
<td>ACR decrease time factor</td>
</tr>
<tr>
<td>ATM</td>
<td>Asynchronous Transfer Mode</td>
</tr>
<tr>
<td>BECN</td>
<td>Backward Explicit Congestion Notification</td>
</tr>
<tr>
<td>BN</td>
<td>BECN (bit in RM-cell)</td>
</tr>
<tr>
<td>BRM</td>
<td>Backward RM-cell</td>
</tr>
<tr>
<td>BT</td>
<td>Burst Tolerance</td>
</tr>
<tr>
<td>CAC</td>
<td>Connection Admission Control</td>
</tr>
<tr>
<td>CBR</td>
<td>Constant Bit Rate</td>
</tr>
<tr>
<td>CCR</td>
<td>Current Cell Rate</td>
</tr>
<tr>
<td>CDF</td>
<td>Cut off decrease factor</td>
</tr>
<tr>
<td>CDV</td>
<td>Cell Delay Variation</td>
</tr>
<tr>
<td>CDVT</td>
<td>CDV Tolerance</td>
</tr>
<tr>
<td>CI</td>
<td>Congestion Indication (bit in RM-cell)</td>
</tr>
<tr>
<td>CLP</td>
<td>Cell Loss Priority</td>
</tr>
<tr>
<td>CLR</td>
<td>Cell Loss Ratio</td>
</tr>
<tr>
<td>CRM</td>
<td>Missing RM cell count</td>
</tr>
<tr>
<td>CTD</td>
<td>Cell Transfer Delay</td>
</tr>
<tr>
<td>DES</td>
<td>Destination End-System</td>
</tr>
<tr>
<td>DGCRA</td>
<td>Dynamic GCRA</td>
</tr>
</tbody>
</table>
DIR  Direction (bit in RM-cell)
EFCI  Explicit Forward Congestion Indication
ER  Explicit Rate
FECN  Forward Explicit Congestion Notification
FRTT  Fixed round trip time
GCRA  Generic Cell Rate Algorithm
GFC  Generic Flow Control
ICR  Initial Cell Rate
LVST  Last Virtual Schedule Time
maxCTD  Maximum Cell Transfer Delay
MCR  Minimum Cell Rate
MCRmin  Minimum acceptable MCR
Mrm  Minimum number of cells between RM-cell generation
NI  No Increase (bit in RM-cell)
NNI  network to node interface
Nrm  Maximum number of cells between RM-cell generation
nrt-VBR  Non-Real-Time VBR
OAM  Operations, Administration and Maintenance
PCR  Peak Cell Rate
PACR  Potential Allowed Cell Rate
PDU  Protocol Data Unit
pk-to-pk CDV  Peak-to-peak Cell Delay Variation
QL  Queue Length (field in RM-cell)
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RA</td>
<td>Resource Allocation (bit in RM-cell)</td>
</tr>
<tr>
<td>RDF</td>
<td>Rate Decrease Factor</td>
</tr>
<tr>
<td>RIF</td>
<td>Rate Increase Factor</td>
</tr>
<tr>
<td>RM-cell</td>
<td>Resource Management Cell</td>
</tr>
<tr>
<td>RSC</td>
<td>Rate Sequence Check</td>
</tr>
<tr>
<td>RTD</td>
<td>Round-trip delay</td>
</tr>
<tr>
<td>RTT</td>
<td>Round-trip Time</td>
</tr>
<tr>
<td>rt-VBR</td>
<td>Real-Time VBR</td>
</tr>
<tr>
<td>SCR</td>
<td>Sustainable Cell Rate</td>
</tr>
<tr>
<td>SES</td>
<td>Source End-System</td>
</tr>
<tr>
<td>SN</td>
<td>Sequence Number (field in RM-cell)</td>
</tr>
<tr>
<td>τ₂</td>
<td>Upper bound round trip time</td>
</tr>
<tr>
<td>τ₃</td>
<td>Lower bound round trip time</td>
</tr>
<tr>
<td>TAT</td>
<td>Theoretical Arrival Time</td>
</tr>
<tr>
<td>TBE</td>
<td>Transient buffer exposure</td>
</tr>
<tr>
<td>TCR</td>
<td>Tagged Cell Rate</td>
</tr>
<tr>
<td>TM</td>
<td>Traffic Management</td>
</tr>
<tr>
<td>Trm</td>
<td>It is the upper bound on the time between successive forward RM cells</td>
</tr>
<tr>
<td>UBR</td>
<td>Unspecified Bit Rate</td>
</tr>
<tr>
<td>UNI</td>
<td>User network Interface</td>
</tr>
<tr>
<td>UPC</td>
<td>Usage Parameter Control</td>
</tr>
<tr>
<td>VBR</td>
<td>Variable Bit Rate</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>VC</td>
<td>Virtual Connection</td>
</tr>
<tr>
<td>VCC</td>
<td>Virtual Channel Connection</td>
</tr>
<tr>
<td>VCI</td>
<td>Virtual Channel Identifier</td>
</tr>
<tr>
<td>VPC</td>
<td>Virtual Path Connection</td>
</tr>
<tr>
<td>VPI</td>
<td>Virtual Path Identifier</td>
</tr>
<tr>
<td>VS/VD</td>
<td>Virtual Source/Virtual Destination</td>
</tr>
</tbody>
</table>
List of Figures

Figure 2.2.1       The ATM cell format  8
Figure 3.3.1.1    Generic Cell Rate Algorithm (GCRA)  32
Figure 3.3.2      Police rate envelope - Algorithm A  38
Figure 3.3.3      Police rate envelope - Algorithm B (2_Store)  42
Figure 3.3.4      Police rate envelope – Cyclic N_Store algorithm  47
Figure 3.3.4a     Simplified flow chart of Cyclic N_Store algorithm  48
Figure 4.2.1a     Police rate envelope vs ER arrival (at policer) rate envelope of Algorithm A (τ_2 = 5 ms and τ_3 = 1 ms)  60
Figure 4.2.1b     Police rate envelope vs ER arrival (at policer) rate envelope of Algorithm B (2_Store, τ_2 = 5 ms and τ_3 = 1 ms)  60
Figure 4.2.1c     Police rate envelope vs ER arrival (at policer) rate envelope of Cyclic N_Store (τ_2 = 5 ms and τ_3 = 1 ms)  61
Figure 4.2.1d     Police rate envelope vs ER arrival (at policer) rate envelope Cyclic N_Store with CORRECTION 1 and 2 (τ_2 = 5 ms and τ_3 = 1 ms)  61
Figure 4.3.1.1    End to end communication between actual source and destination  67
Figure 4.3.1.2    End to end communication between virtual source and destination  67
Figure 5.1.1      A model of UNI with an ABR policer  79
Figure 5.1.6.1    1 Hop Reference Network Model  88
Figure 5.1.6.1    2 Hop Reference Network Model  88
Figure 5.2.1      N_Store_I Algorithm-Effect of N over number of police rates-WAN Environment 92
Figure 5.2.2      N_Store_I Algorithm-Effect of N over percentage of excess cells-WAN Environment 92
Figure 5.2.3  N_Store_I Algorithm-Effect of N over percentage of tagged cells-
WAN Environment 92

Figure 5.3.1  Performance of policing in terms of percentage Tagged cells in
LAN environment with RTD varying in accordance with VBR 
traffic variations ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to
1000 micro secs.) 94

Figure 5.3.2  Performance of policing in terms of percentage Tagged cells in
MAN environment with RTD varying in accordance with VBR 
traffic variations ($\tau_3 = 1000$ micro secs. and $\tau_2$ = varied from 1000
to 5000 micro secs.) 94

Figure 5.3.3  Performance of policing in terms of percentage Tagged cells in
WAN environment with RTD varying in accordance with VBR 
traffic variations ($\tau_3 = 5000$ micro secs. and $\tau_2$ = varied from 5000
to 15000 micro secs.) 94

Figure 5.3.1.1  Envelope process – Nothing scheduled 95

Figure 5.3.4  Cyclic N_Store with error corrections- Police rate envelope
($\tau_2$ = 5 ms and $\tau_3$ = 1 ms) 100

Figure 5.3.2.1  Performance of policing in terms of percentage Tagged cells in
LAN environment with randomly varied RTD ($\tau_3 = 100$ micro 
secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 103

Figure 5.3.2.2  Performance of policing in terms of percentage Excess cells in
LAN environment with randomly varied RTD ($\tau_3 = 100$ micro 
secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 103

Figure 5.3.2.3  Performance of policing in terms of percentage Police rates in
LAN environment with randomly varied RTD ($\tau_3 = 100$ micro 
secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 103

Figure 5.3.2.4  Performance of policing in terms of percentage Tagged cells in
LAN environment with randomly varied (Ramping up and down) 
RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro 
secs.) 104

Figure 5.3.2.5  Performance of policing in terms of percentage Excess cells in
LAN environment with randomly varied (Ramping up and down) 
RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro 
secs.) 104
Figure 5.3.2.6 Performance of policing in terms of percentage Police rates in LAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 104

Figure 5.3.2.7 Performance of policing in terms of percentage Tagged cells in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 105

Figure 5.3.2.8 Performance of policing in terms of percentage Excess cells in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 105

Figure 5.3.2.9 Performance of policing in terms of percentage Police rates in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 105

Figure 5.3.2.10 Performance of policing in terms of percentage Tagged cells in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 108

Figure 5.3.2.11 Performance of policing in terms of percentage Excess cells in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 108

Figure 5.3.2.12 Performance of policing in terms of percentage Police rates in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 108

Figure 5.3.2.13 Performance of policing in terms of percentage Tagged cells in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 109

Figure 5.3.2.14 Performance of policing in terms of percentage Excess cells in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 109

Figure 5.3.2.15 Performance of policing in terms of percentage Police rates in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 109
Figure 5.3.2.16  Performance of policing in terms of percentage Tagged cells in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 110

Figure 5.3.2.17  Performance of policing in terms of percentage Excess cells in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 110

Figure 5.3.2.18  Performance of policing in terms of percentage Police rates in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 110

Figure 5.3.2.19  Performance of policing in terms of percentage Tagged cells in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 113

Figure 5.3.2.20  Performance of policing in terms of percentage Excess cells in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 113

Figure 5.3.2.21  Performance of policing in terms of percentage Police rates in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 113

Figure 5.3.2.22  Performance of policing in terms of percentage Tagged cells in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 114

Figure 5.3.2.23  Performance of policing in terms of percentage Excess cells in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 114

Figure 5.3.2.24  Performance of policing in terms of percentage Police rates in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 114

Figure 5.3.2.25  Performance of policing in terms of percentage Tagged cells in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2$ = varied from 100 to 1000 micro secs.) 115
Figure 5.3.2.26  Performance of policing in terms of percentage Excess cells in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs}$.) 115

Figure 5.3.2.27  Performance of policing in terms of percentage Police rates in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs}$.) 115

Figure 6.4.1  Performance of policing in terms of percentage Tagged cells in LAN environment with randomly varied RTD ($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 125

Figure 6.4.2  Performance of policing in terms of percentage Excess cells in LAN environment with randomly varied RTD ($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 125

Figure 6.4.3  Performance of policing in terms of percentage Police rates in LAN environment with randomly varied RTD ($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 125

Figure 6.4.4  Performance of policing in terms of percentage Tagged cells in LAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 126

Figure 6.4.5  Performance of policing in terms of percentage Excess cells in LAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 126

Figure 6.4.6  Performance of policing in terms of percentage Police rates in LAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 126

Figure 6.4.7  Performance of policing in terms of percentage Tagged cells in LAN environment with RTD varying in accordance with VBR traffic variation($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 127

Figure 6.4.8  Performance of policing in terms of percentage Excess cells in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = \text{100 micro secs. and } \tau_2 = \text{varied from 100 to 1000 micro secs}$.) 127
Figure 6.4.9  Performance of policing in terms of percentage Police rates in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 127

Figure 6.4.10  Performance of policing in terms of percentage Tagged cells in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 129

Figure 6.4.11  Performance of policing in terms of percentage Excess cells in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 129

Figure 6.4.12  Performance of policing in terms of percentage Police rates in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 129

Figure 6.4.13  Performance of policing in terms of percentage Tagged cells in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 130

Figure 6.4.14  Performance of policing in terms of percentage Excess cells in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 130

Figure 6.4.15  Performance of policing in terms of percentage Police rates in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 130

Figure 6.4.16  Performance of policing in terms of percentage Tagged cells in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 131

Figure 6.4.17  Performance of policing in terms of percentage Excess cells in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 131

Figure 6.4.18  Performance of policing in terms of percentage Police rates in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 =$ varied from 100 to 1000 micro secs.) 131
Figure 6.4.19  Performance of policing in terms of percentage Tagged cells in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 133

Figure 6.4.20  Performance of policing in terms of percentage Excess cells in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 133

Figure 6.4.21  Performance of policing in terms of percentage Police rates in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 133

Figure 6.4.22  Performance of policing in terms of percentage Tagged cells in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 134

Figure 6.4.23  Performance of policing in terms of percentage Excess cells in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 134

Figure 6.4.24  Performance of policing in terms of percentage Police rates in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 134

Figure 6.4.25  Performance of policing in terms of percentage Tagged cells in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 135

Figure 6.4.26  Performance of policing in terms of percentage Excess cells in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 135

Figure 6.4.27  Performance of policing in terms of percentage Police rates in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 135

Figure 6.5.1  Simplified flow chart of Improved Cyclic N_Store_1 algorithm 140
Figure 6.5.1.1 Performance of policing in terms of percentage Tagged cells in LAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 142

Figure 6.5.1.2 Performance of policing in terms of percentage Excess cells in LAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 142

Figure 6.5.1.3 Performance of policing in terms of percentage Police rates in LAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 142

Figure 6.5.1.4 Performance of policing in terms of percentage Tagged cells in LAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 143

Figure 6.5.1.5 Performance of policing in terms of percentage Excess cells in LAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 143

Figure 6.5.1.6 Performance of policing in terms of percentage Police rates in LAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 143

Figure 6.5.1.7 Performance of policing in terms of percentage Tagged cells in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 144

Figure 6.5.1.8 Performance of policing in terms of percentage Excess cells in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 144

Figure 6.5.1.9 Performance of policing in terms of percentage Police rates in LAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 144

Figure 6.5.1.10 Performance of policing in terms of percentage Tagged cells in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 146
Figure 6.5.1.11  Performance of policing in terms of percentage Excess cells in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 146

Figure 6.5.1.12  Performance of policing in terms of percentage Police rates in MAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 146

Figure 6.5.1.13  Performance of policing in terms of percentage Tagged cells in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 147

Figure 6.5.1.14  Performance of policing in terms of percentage Excess cells in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 147

Figure 6.5.1.15  Performance of policing in terms of percentage Police rates in MAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 147

Figure 6.5.1.16  Performance of policing in terms of percentage Tagged cells in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 148

Figure 6.5.1.17  Performance of policing in terms of percentage Excess cells in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 148

Figure 6.5.1.18  Performance of policing in terms of percentage Police rates in MAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 148

Figure 6.5.1.19  Performance of policing in terms of percentage Tagged cells in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 150

Figure 6.5.1.20  Performance of policing in terms of percentage Excess cells in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from} 100 \text{ to } 1000$ micro secs.) 150
Figure 6.5.1.21 Performance of policing in terms of percentage Police rates in WAN environment with randomly varied RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 150

Figure 6.5.1.22 Performance of policing in terms of percentage Tagged cells in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 151

Figure 6.5.1.23 Performance of policing in terms of percentage Excess cells in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 151

Figure 6.5.1.24 Performance of policing in terms of percentage Police rates in WAN environment with randomly varied (Ramping up and down) RTD ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 151

Figure 6.5.1.25 Performance of policing in terms of percentage Tagged cells in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 152

Figure 6.5.1.26 Performance of policing in terms of percentage Excess cells in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 152

Figure 6.5.1.27 Performance of policing in terms of percentage Police rates in WAN environment with RTD varying in accordance with VBR traffic variation ($\tau_3 = 100$ micro secs. and $\tau_2 = \text{varied from 100 to 1000 micro secs.}$) 152

Figure AP2.a Cyclic N_Store Algorithm – Reception Process 168

Figure AP2.b Cyclic N_Store Algorithm – Envelope Process – Rate increase 169

Figure AP2.c Cyclic N_Store Algorithm – Envelope Process – Nothing Scheduled 170

Figure AP2.d Cyclic N_Store Algorithm – Time Sequence Adjustment (TSA) 171

Figure AP2.e Cyclic N_Store Algorithm – Delivery Process 172

xix
Figure AP2.f  Cyclic N_Store Algorithm – Pointer Process 173
Figure AP2.g  Cyclic N_Store Algorithm – Rate Sequence Check (RSC) 174
Figure AP3.a  Cyclic N_Store_I Algorithm – Reception Process 180
Figure AP3.b  Cyclic N_Store_I Algorithm – Envelope Process – Nothing Scheduled 181
Figure AP3.c  Cyclic N_Store_I Algorithm – Time Sequence Adjustment (TSA) 182
Figure AP3.d  Cyclic N_Store_I Algorithm – Delivery Process 183
Figure AP3.e  Cyclic N_Store_I Algorithm – Pointer Process 184
Figure AP3.f  Cyclic N_Store_I Algorithm – Rate Sequence Check (RSC) 185
Figure AP4.1  RTT estimation process 191
List of Tables

Table 2.2.1 The ATM cell format 8
Table 2.4.1 The QoS Parameters (Reproduced from TM spec. [1]) 13
Table 3.3.2 Police rates and their schedules evaluated by Algorithm A 37
Table 3.3.2a List of police rates and their schedules delivered by Algorithm A 38
Table 3.3.3 Police rates and their schedules evaluated by Algorithm B (2_Store) 41
Table 3.3.3a List of police rates and their schedules delivered by Algorithm B (2_Store) 42
Table 3.3.4a N_Store police rate table built by ‘Nothing scheduled routine of envelope process which introduces ERROR 1 46
Table 3.3.4b N_Store police rate table 3.3.4a is adjusted by ‘Time sequence adjustment routine’ (TSA) and ‘Something scheduled’ routine of envelope process 46
Table 3.3.4c N_Store police rate table built by ‘Nothing scheduled routine of envelope process after the adjustment done in table 3.3.4b 46
Table 3.3.4d N_Store police rate table 3.3.4c is adjusted by ‘Time sequence adjustment routine’ (TSA) and ‘Something scheduled’ routine of envelope process which introduces ERROR 2 46
Table 3.3.4e Police rates PR0 to PR14 and their schedules determined by Cyclic N_Store 47
Table 3.3.4f Police rates and their schedules stored by Algorithm A, 2_Store and N_Store 53
Table 4.2 Percentage of rates delivered by N_Store algorithm 56
Table 4.3 The ABR service parameters 69
Table 4.3.5.1 The RM cell field settings 72
Table 5.2.1  The typical parameter values used in the simulation of network with the ABR policer  90

Table 5.3.1  Police rates and their schedules stored by N_Store before and after the correction of ERROR 1  96

Table 5.3.2  N_Store police rate table ‘Nothing scheduled’ and ‘Something scheduled’ routine of envelope process after the adjustment done in table 3.3.4d that introduced the ERROR 2  97

Table 5.3.2a  Police rates PR0 to PR16 and their schedules determined by Cyclic N_Store 98

Table 6.2.1  Police rates and their schedules stored by N_Store after the improvement 122

Table 6.5.1  Police rates and their schedules determined by N_Store_1 138

Table AP1.1  The RM cell structure 165

Table AP1.2  The RM cell field size 165

Table AP1.3  The RM cell field setting 165
Table of contents

Abstract i

Statement of Originality iv

Acknowledgments v

List of Abbreviations vi

List of Figures x

List of Tables xxi

1. Introduction
   1.1 Background 1
   1.2 Thesis Outline 3
   1.3 Contributions 5

2. Traffic and Congestion control in the ABR service of ATM networks
   2.1 Introduction 7
   2.2 ATM network and its service architecture 8
   2.3 Congestion in ATM networks 10
   2.4 Quality of service 12
   2.5 Traffic and Congestion control functions 14
   2.6 Traffic Police and its performance measurement in ATM network 19
   2.7 The ABR police and its performance measurement 24

3. An Overview of the ABR policer
   3.1 Introduction 26
   3.2 Traffic contract and Cell Conformance 27
      3.2.1 Cell conformance and connection compliance 28
      3.2.2 Cell Conformance definition 28
         3.2.2.1 Round Trip Delay (RTD) 29
         3.2.2.2 The ABR conformance definition 30
   3.3 The ABR policing related algorithms 31
      3.3.1 Dynamic Generic Cell Rate Algorithm (DGCRA) 32
         3.3.1.1 DGCRA using leaky bucket algorithm 32
         3.3.1.2 DGCRA using virtual scheduling algorithm 34
         3.3.1.3 Determination of police rate 1(k) and its schedule 35
4. ABR policer performance measurement and protocol operation
   4.1 Introduction 54
   4.2 A review of the performance measurement of an ABR policer 55
      4.2.1 A new method of performance measurement through Excess and
             Tagged cells 58
   4.3 The ABR policer protocol operation 66
      4.3.1 The ABR flow control 66
      4.3.2 Data cell and RM cell type, structure & its fields 68
      4.3.3 ABR service parameters and negotiation 68
      4.3.4 The Source, Destination and Switch behaviours 70
      4.3.5 The ABR flow control process 71
         4.3.5.1 End system – Sending forward RM cells, Backward RM Cells
                  and forward Data Cells. 72
         4.3.5.2 End system – Receiving forward RM cells, Backward RM
                  Cells and forward Data Cells. 75

5. ABR policer simulation
   5.1 Introduction 77
      5.1.1 User network interface model 78
      5.1.2 Source model 80
      5.1.3 Switch model 81
      5.1.4 Destination model 83
      5.1.5 ABR policer model 83
      5.1.6 RTD Variation and choice of $\tau_2$ and $\tau_3$ 84
   5.2 Assumptions and Parameter values 89
   5.3 Results 93
      5.3.1 Result analysis and errors in N_Store algorithm 93
         5.3.1.1 ERROR 1 95
         5.3.1.2 ERROR 2 97
         5.3.1.3 CORRECTION 1 99
         5.3.1.4 CORRECTION 2 99
      5.3.2 Performance results 101
         5.3.2.1 LAN environment 101
         5.3.2.2 MAN environment 106
         5.3.2.3 WAN environment 111
   5.4 Conclusions 117

6. Improved N_Store algorithm and its performance review
   6.1 Introduction 119
   6.2 Inaccuracy in N_Store Algorithm 120
   6.3 Rate Sequence Check and accurate N_Store Algorithm 122
   6.4 Results 124
6.4.1 Performance results 124
  6.4.1.1 LAN environment 124
  6.4.1.2 MAN environment 128
  6.4.1.3 WAN environment 132
6.5 Simplified N_Store_I 136
  6.5.1 Performance results 141
  6.5.1.1 LAN environment 141
  6.5.1.2 MAN environment 145
  6.5.1.3 WAN environment 149
6.6 Conclusions 153

7. Conclusions
  7.1 Overview 154
  7.2 Future work 159

References 160

Appendix 1: RM cell structure 165

Appendix 2: Cyclic N_Store Algorithm
  2.1 Flow Chart 168
  2.2 Pseudo Code 175

Appendix 3: Cyclic N_Store_I Algorithm
  3.1 Flow Chart 180
  3.2 Pseudo Code 186

Appendix 4: RTT estimation 2 Pass Method 190
1. Introduction

1.1 Background

The ATM network will be the future generation of high-speed networks, capable of handling various traffic types eg: data, voice and video and, in particular, traffic with real time constraints. The Available Bit Rate (ABR) and Unspecified Bit Rate (UBR) services have been introduced by the ATM Forum to cater for traffic from applications such as e-mail, news feed and file transfer that does not require a delay guarantee. In particular, the ABR service has been designed to cater for traffic sources that demand low loss ratios but can accept longer delays. The ABR utilises only the bandwidth unused by VBR and CBR. This makes the ABR bandwidth vary continuously throughout the connection time. In order to minimise the cell-loss under such environment, a rate based closed loop feedback mechanism is adopted. This mechanism involves two feedback modes: Binary and Explicit Rate (ER) modes.

The ATM Forum has defined a complex set of rules to define how the ABR source should behave in controlling its rate upon reception of feedback from the network. The source is assured of low cell-loss (Quantitative assurance of Cell loss ratio is network specific) if it strictly follows the set behaviour. If many ABR users are sharing a common link then all users are required to follow their respective set behaviour. Any misbehaviour from a user will affect the rest in terms of fair allocation of the bandwidth and minimum cell-
loss guarantee (if provided by the network). It is found that ABR, while more complex to implement, performs better than UBR only in controlling the fairness between the VCs and reducing less loss of cells during congestion [17]. However the performance of the ABR service depends on the source behaviour. One or more misbehaving sources can cause this performance to deteriorate significantly. If these misbehaving sources are not tracked and punished, the intended purpose of the ABR service to guarantee a low cell loss may not be obtained. This control is achieved through a traffic contract, agreed upon between user and network, and a policer to monitor the source traffic and tag the cells for rate violations. Thus, in addition to CAC, the policer is another control point for maintaining the performance of the ABR service. However, an inaccurate policer can significantly affect the ABR performance.

The ATM TM Specification has defined DGCRA to check the conformance of a cell received from each source with respect to a reference feedback rate (police rate) evaluated using Algorithm A or B [1]. The performance of a policer depends on the performance of the above algorithms. Algorithm A or B evaluates a police rate and schedules it for enforcing. The scheduling is done as per the conformance definition [1]. If the algorithm cannot accurately determine the police rate and its schedule then Type 1 and Type 2 errors may occur leading to tagged and excess cells. Thus the performance of an algorithm can be quantified in terms of tagged cells and excess cells.

If a policing algorithm meets the requirement of the conformance definition, then it should not tag a cell from a complaint connection. However this will not be known unless the performance is measured by quantifying the tagged cells that an algorithm produces. So far, no such performance measurements have been carried out and without it the effectiveness of a policer will not be known.
The proposal of such a method used in quantitative performance measurement of different policer algorithms is the main topic of this thesis. As an outcome of the performance measurement errors are identified in the N_Store algorithm, corrections are suggested, and a new simplified N_Store_I algorithm is introduced.

1.2 Thesis Outline:

The thesis is organised into 6 chapters with 5 Appendices.

Chapter 2 presents an overview of the ATM network service architecture, congestion in ATM networks and various traffic control functions, which are applied to avoid congestion. It focuses further on the ABR service and its flow control mechanism. The chapter then discusses the need for policing and its key role in traffic management. Following this, the literature on the performance measurement of policing is reviewed. The aim of the review is to highlight the significance of the performance measurement of a policer.

Chapter 3 reviews the ABR policer and algorithms DGCRA, A, B(2_Store) and N_Store used to implement them. The basic principle with which a policer operates depends on the nature of the traffic contract made with the user and the conformance definition. The traffic contract and conformance definition specified by the ATM Forum for the ABR service is discussed in detail. The ABR policer related algorithms specified in TM 4.0 [1] are also described.

Chapter 4 introduces a new method for measuring the performance of the ABR policer. In this new method, the effectiveness of a policer is quantified in terms of tagged cells and excess cells. A new ABR policer performance model, on which the simulation is based, is presented. A detailed description of ABR flow control is given here. This provides the technical background
needed to understand the simulation model. The performance results obtained in terms of tagged cells and excess cells are analysed. This performance analysis, which is one of the major contributions of this thesis, reveals previously unknown errors and inefficiencies in the N_Store algorithm. The performance of the N_Store algorithm is found to be poor and does not match the performance of Algorithm A.

The chapter presents two errors and an inefficiency identified as a result of this study. Corrections for these two errors are presented in this chapter but efficiency improvement is presented in chapter 5.

Chapter 4 also presents the performance results graphically for each algorithm (A, 2_Store and N_Store) in terms of percentage tagged cells, excess cells and police rates (on the Y axis) for various values τ₂ (on the X axis). The results of N_Store algorithm, without corrections for errors, show abnormal variations and hence cannot be compared with the results of Algorithm A and 2_Store. Hence the chapter first concentrates on identifying the errors and correcting them. The results obtained after the corrections of both the errors show normal variation and are thus ready for comparison.

The comparison shows that the N_Store algorithm is inaccurate compared to Algorithm A and 2_Store. The accuracy is improved by modifying the algorithm that eliminates the inefficiency. This is described in chapter 5 along with the a Rate Sequence Check (RSC) routine that is presented as an improvement to current N_Store algorithm. The performance results of modified N_Store algorithm that include corrections for errors and RSC routine is presented and proved that the modified algorithm produces less cell loss and equivalent to Algorithm A.

A simplified form of modified N_Store algorithm called N_Store_I is presented. It is simplified by reducing the envelope process routine from the
original N_Store algorithm. The performance results are presented to show that the N_Store_I algorithm produces less cell loss and equivalent to Algorithm A.

The thesis examines key aspects for the performance measurement of the ABR policer. The outcomes of this examination are; 1. The proposal of a new method in the performance measurement, 2. Detection of errors in the N_Store algorithm and 3. The proposal of simplified N_Store_I algorithm for the ABR policer. These proposals along with other recommendations are summarised in chapter 6, which also contains suggestions for further areas of research and future work.

1.3 Contributions:

This section lists the works and contributions contained in this thesis, providing section references where the relevant works are discussed.

1. Proposal of a new method in measuring the performance of the ABR policer. It is proposed to measure the effectiveness of an algorithm in performing a policing function in terms of percentage of excess cells, tagged cells and police rates (Chapter 4, section 4.2.2).

2. Development of the ABR policer performance which implements the Algorithms A, B, N_Store, DGCRA and ERICA algorithms accurately (Chapter 4, section 4.4).

3. Demonstration, through the examination of quantitative results, that the performance of N_Store algorithm is poor and does not match with the performance of Algorithm A. Hence it can not be used in place of Algorithm A (Chapter 4, section 4.6).

4. Discovery of two errors and one inefficiency in N_Store algorithm which are the causes of its poor performance. Corrections are suggested to rectify the errors and improvements are suggested to overcome the
inefficiencies. It is demonstrated through simulation results that the performance improved sufficiently to match the Algorithm A (Chapter 4 section 4.6 and chapter 5 sections 5.2, 5.3 and 5.4)

5. Development of a new N_Store_I algorithm which is a simplified version of N_Store. This N_Store_I algorithm is recommended as being more suitable for implementing the ABR policer due to its simple structure and the best performance that is equivalent to Algorithm A (Chapter 5, section 5.5)
2. Traffic and Congestion control in the ABR service of ATM networks

2.1 Introduction

This chapter reviews the ATM network service architecture, congestion in ATM networks and the traffic and congestion control mechanisms. The thesis is mainly concerned with the ABR service. Hence the greater emphasis on the ABR service and its traffic and congestion control mechanisms. The ABR policer which forms a part of traffic control is described in detail in chapter 3.

Section 2.2 gives a brief background of the ATM networks and their various service categories, with an emphasis on the ABR service. Section 2.3 describes the congestion and its control in ATM networks. Section 2.4 describes the objectives of Quality of Services (QoS) offered to users, which forms the basis for the control mechanisms. Section 2.5 gives an overview of various traffic and congestion control mechanisms. Section 2.6 discusses in detail of the ABR flow control and provides the technical background for the ABR policing mechanisms and the simulation model described in Chapter 3 and 5 respectively.
2.2 ATM network and service architecture

Asynchronous Transfer Mode (ATM), which is a high speed network, transmits information using short fixed-size cells consisting of 48 bytes of payload and 5 bytes of header as shown in fig. 2.2.1. The fixed size of the cells reduces the delay variance making the networks suitable for integrated traffic consisting of voice, video and data.

<table>
<thead>
<tr>
<th>Header 5 Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GFC (4 bits)</td>
</tr>
<tr>
<td>VPI</td>
</tr>
<tr>
<td>VCI</td>
</tr>
<tr>
<td>VCI</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Fig 2.2.1: The ATM cell format

Generic Flow control (GFC) was intended on a flow control mechanism, but this has not been used. The Virtual Path Identifier (VPI) therefore used for addressing the Virtual Path (VP) within ATM network. Virtual channels (VCs) are the basic paths for data transportation. Each user is assigned with a VC. Many VCs are transported within VPs, essentially form a data pipe. The Virtual Channel Identifier (VCI) is used for addressing individual VCs. Payload type indicator (PTI) identifies the type of data being carried by the cell. Cell loss priority (CLP) used to identify a cell as either higher or lower priority one. CLP=1 indicates a low priority, whereas CLP=0 indicates a
high priority. The network during congestion may drop low priority cells. This also helps in managing the traffic differently for high and low priority cells. Header Error Correction (HEC) is the checksum for the first 4 bytes of the header. The HEC helps the receiver to confirm the validity of the cell contents.

ATM networks establish a virtual circuit between two end systems through the network. The services requirements and traffic parameters of the connection are indicated to the intermediate switches and the destination before establishing the connection. During connection setup the users submit their service requirements and traffic parameters. The network along with the destination negotiates the requirements. The connection is established once the user agrees to condition, which may include certain traffic parameters as demanded by the network in order to meet the required quality of service.

There are 5 service categories provided at the ATM layer. They are, Constant Bit Rate (CBR), Variable Bit Rate Real-Time (rt-VBR), Non-Real-Time (nrt-VBR), Unspecified Bit Rate (UBR) and Available Bit Rate (ABR).

Applications such as voice telephony and video requires a small delay variation throughout the connection. To enable this, the connection demands a fixed bandwidth. The CBR category provides such a fixed bandwidth connection. The maximum bandwidth is characterised by Peak Cell Rate (PCR).

The real-time VBR (rt-VBR) service category is devised for real-time applications (interactive video) that are delay sensitive, similar to CBR. The non-real-time VBR (nrt-VBR) service category is devised for applications that are not sensitive to delay, such as multimedia email, which have bursty traffic characteristics. The rt-VBR and nrt-VBR connections are characterised in terms of a Peak Cell Rate (PCR), Maximum Burst Size (MBS), and Sustainable Cell Rate (SCR).
The UBR service is devised for those applications, which are not sensitive to delay and cell loss. In the UBR category, only the leftover bandwidth on a link is available. No traffic related service (delay or cell loss) guarantees are provided. The main applications are e-mail and file transfer.

The ABR service category is also designed for applications such as e-mail and file transfer, but for the user who demands low cell loss, while accepting variable delays. Unlike UBR, the ABR service is loss sensitive. ABR uses the bandwidth that is unused by other VBR and CBR services. The ABR users have to share the bandwidth dynamically (based on the feedback) with other users including CBR and VBR which share the common link. A low cell loss guarantee will be provided only to those users who strictly follow the set behaviour for the source. Networks may monitor the ABR source behaviour by deploying a policer at the user and network interface.

### 2.3. Congestion in ATM networks

The ATM network is envisaged as a high speed network capable of carrying various traffic such as voice and video that are delay sensitive as well as bursty traffic such as e-mail and file transfers which are loss sensitive. Managing such real time and non-real time traffic to provide the required quality of service is very complex. Lack of proper feedback of control information has constrained the control of the flow of user cells. Uncontrolled traffic flow may lead to network congestion. The ATM network multiplexes various traffic types and transports them via switch nodes. Data traffic because of its bursty nature, may occupy a large bandwidth for a short time, and then lie idle for a long time. This makes the traffic unpredictable at switching node. The network, without predicting the load, may not reserve extra resources in advance. Under heavy load conditions if many data bursts from different sources arrive simultaneously
at a node, very large queues may develop resulting in buffer overflow and lead to congestion. Similarly the congestion can also occur if the high-speed links are introduced in slower networks [4]. Under such scenarios the total input rate may exceed the output link capacity leading to congestion in ATM networks. The ATM congestion and its control mechanisms have been reviewed in the literatures [4],[32],[33],[34],[35], [38],[40],[42],[45],[47] are presented below.

Congestion means a network resource wasting process (cell loss and retransmission) caused by overload and congestion control results in the control of resource wastage without restricting the access to the resources. The author [42] has precisely defined the congestion in ATM networks and presented a comparative discussion of various control methods in terms of control motivation, control power, multiplexing gain and cost.

In [40] the author defines network congestion as in ITU-TS as a state where network elements are unable to maintain a negotiated QoS on existing connections or new connections request. Based on this the author reviews the CAC and UPC scheme.

When the congestion occurs at any node point in the ATM network, the throughput falls and the response time is raised drastically with the network load [47]. The author in [47] has described that the congestion avoidance scheme avoids network becoming congested at knee point and a proper congestion control scheme prevent the drastic fall of throughput from the cliff after the congestion has occurred. The author goes on to demonstrate through congestion avoidance mechanisms, that the network will operate in the optimal region of low delay and high throughput.

The author in [45] has described how congestion occurs when the demand exceeds the availability of resources. However it is not a resource shortage problem but a dynamic problem. It cannot be solved by static solutions such as increasing buffer size, matching the link speed and increasing the
processor speed. The author suggests that the network be protected in the event of congestion through suitable protocols. Congestion schemes are applied either to reduce the demand or to increase the resources.

The congestion in ATM LAN and MAN are described in [33] and [34] and congestion in ABR service of ATM networks is described in [32] and [35].

To avoid network congestion, suitable traffic management and control mechanisms are needed. These allow the network to operate more efficiently, and ensure good quality service under increasing traffic demands. The various traffic control methods adopted depend on the type of QoS agreed to the user. Hence understanding QoS, and commitments made through a set of parameters by various service categories, are essential. These topics are described in the next section.

2.4 Quality of service (QoS)

The Quality of Service (QoS) offered by the ATM network is measured in terms of ATM connection parameters.

The TM specification [1] has identified 6 QoS parameters. The network may offer one or more of these parameters for each connection. Such an offer depends on the performance objectives supported by the network. The performance objective of an individual network is achieved by implementing network specific control mechanisms.

Among the 6 parameters only 3 are negotiated. The list of negotiated and non-negotiated parameters is given below. The TM specification [1] gives the description of these parameters.

Negotiable Parameters are:

Peak-to-peak Cell Delay Variation (peak-to-peak CDV)
Maximum Cell Transfer Delay (\(\text{maxCTD}\))

Cell Loss Ratio (CLR) = \(\frac{\text{Lost cells}}{\text{Total Transmitted cells}}\)

Non-negotiable Parameters are:

Cell Error Ratio (CER)

Severely Errored Cell Block Ratio (SECBR)

Cell Misinsertion Rate (CMR)

The QoS requirements of each of 5 ATM network service categories (section 2.2) are related to network behaviour. The control functions such as CAC (call admission control), routing and resource allocation are different for each service category. Thus the QoS guarantee, in terms of negotiable parameters for each category will be either specified or unspecified. Table 2.4.1 shows the QoS parameter being specified or unspecified against each service category.

<table>
<thead>
<tr>
<th>QoS Parameters</th>
<th>CBR</th>
<th>Rt-VBR</th>
<th>Nrt-VBR</th>
<th>UBR</th>
<th>ABR</th>
</tr>
</thead>
<tbody>
<tr>
<td>pk-to-pk CDV</td>
<td>specified</td>
<td>specified</td>
<td>unspecified</td>
<td>unspecified</td>
<td>unspecified</td>
</tr>
<tr>
<td>maxCTD</td>
<td>specified</td>
<td>specified</td>
<td>unspecified</td>
<td>unspecified</td>
<td>unspecified</td>
</tr>
<tr>
<td>CLR</td>
<td>specified</td>
<td>specified</td>
<td>specified</td>
<td>unspecified</td>
<td>specified</td>
</tr>
</tbody>
</table>

Table 2.4.1: The QoS Parameters (Reproduced from TM spec. [1])

The network makes a commitment to guarantee the QoS agreed upon with the user who reserves the resource through any of the 5 service categories (except UBR). This guarantee is, provided to only those connections whose cells conform to the relevant conformance test. Hence the network has to adopt suitable traffic and congestion control methods to ensure the QoS commitments. A basic relationship must exist between cell conformance, traffic control and cell losses in order to define and fulfil the QoS commitment [36]. The author has reviewed the Cell Loss Ratio (CLR) a QoS parameter in relation to the cell conformance definition and UPC/NPC traffic control mechanisms. As a result the author proposes dimensioning of network elements which are located downstream of UPC/NPC mechanisms.
Traffic and Congestion control in the ABR Service of ATM networks

on traffic assumptions based on user traffic contract parameters such as CLR, PCR etc. Traffic and Congestion control functions available to the network are given in the next section.

2.5 Traffic and Congestion control functions

The traffic control function is the set of mechanisms applied by the network to avoid congestion, whereas the congestion control function refers to the set of actions taken when congestion occurs. The actions are mainly taken to minimise the intensity, spread and duration of the congestion [5].

The basic criteria for traffic control is to determine whether a new connection can be admitted and the agreed performance level in terms of QoS parameters can be supported. This requires the subscriber and the network service provider to enter into a traffic contract in which the network service provider agrees to provide a certain level of QoS and the user agrees to follow the behaviour required and not to exceed the specified limits. Traffic control functions based on the contract establish a set of traffic parameters and enforce them on the users. In this way congestion is avoided. If traffic control fails and congestion occurs, then congestion control functions are invoked to respond and recover from the congestion. The traffic and congestion functions specified in TM 4.0 [1] are described below.

The Traffic Control Functions are referred to as Connection Admission Control (CAC), Usage Parameter Control (UPC), Traffic Shaping and Resource Management using Virtual Paths. The Congestion Control Functions are Selective Cell Discarding, Frame Discard, Generic Flow Control (GFC), Explicit Forward Congestion Indication (EFCI) and the ABR Flow Control. A brief description of these control functions is given below.

The ATM networks may implement one or more of the above control functions in order to meet the QoS objectives. CAC accepts a new
connection if the path is not fully loaded but rejects a connection if all paths are fully loaded. This avoids sporadic congestions. A connection request is accepted only when sufficient resources are available at all successive network nodes such that the QoS for the existing connections is maintained. The CAC function will determine whether the connection can be accepted or not based on the traffic contract and the network's definition of a compliant connection. Once the connection is established the CAC provides all the connection traffic parameters needed by Usage Parameter Control (UPC) function generally referred as a policer.

The UPC function monitors the traffic for any violation of contract and controls the flow of non-conforming traffic into the network. Its main purpose is to protect network resources from both malicious as well as unintentional misbehaviour, which can affect the QoS of other connections. This function may be used for policing the source traffic. The UPC/policer detects violations of negotiated parameters and takes appropriate actions, through either tagging or discarding non-conforming cells. The UPC is referred to the connection being monitored at a private or public UNI (User to network interface). The NPC is referred to the connection being monitored at a private or public NNI (network to network Interface). UPC is applied to both user connections and signalling channels. A network's fundamental obligation is to support QoS commitments on compliant connections. Therefore, the UPC/policer must be capable of enforcing traffic contracts.

The UPC/policer monitors the VCCs and VPCs by checking the validity of VPI and the VCI addresses and any violation of agreed traffic parameters. The actions of a policer on the monitored cells include cell passing for conformed cells and cell tagging or discarding for non-conformed cells. The tagging of CLP=0 cells is done by overwriting the CLP bit to 1 (CLP=1). Network resources are allocated separately to CLP=0 and CLP=1 traffic flows and thus the network may provide separate QoS parameters for CLP=0
Traffic and Congestion control in the ABR Service of ATM networks

and CLP=0+1 cell flows. If the network decides not to allocate additional network resources, the CLP=1 cells may be discarded. In this case, tagging is not applicable.

Traffic shaping is used to control the traffic flow to reduce the cell clumping and to ensure the cell conformance at the network interface. A leaky bucket algorithm may be used for traffic shaping. The algorithm monitors the traffic and allows only conforming cells to pass.

During network congestion the non compliant (tagged) CLP=1 cells are discarded to avoid further congestion collapse. This method is called selective cell discard and is adapted mainly so as not to loose CLP=0 cells during a congestion period. However in some cases discarding CLP=1 cells may not be all that effective in congestion control compared to the discard of a frame. Hence Frame discard method is adopted to avoid congestion collapse. The term frame refers to the ATM adaptation layer (AAL) protocol data unit.

The Generic Flow Control (GFC) function refers to the first four bits (refer to fig.1.1) of the cell header. The GFC protocol is discussed in detail in reference [43]. Initially this was envisaged to be used by the network to flow control the source, but later abandoned. At the network to network interface, GFC is not used and the four bits are part of an extended virtual path (VP) field.

Explicit Forward Congestion Indication (EFCI), is a binary means to convey either congestion or no-congestion, and is used by switches adjusting the payload type field in the cell header. End-systems may use the EFCI bit to adaptively control the cell rate, by lowering the rate during congestion and increasing the rate during no-congestion periods. The EFCI bit is cleared in all cells before they leave the source. When the cells pass through a switch the EFCI bit is set to 1 where if there is congestion. The destination will monitor EFCI bits, and accordingly indicates to the source to either increase
or decrease the rate. The performance of EFCI mechanism in the rate adaptive congestion scheme has been applied to ATM LAN [33] and it was found that the throughput increases significantly under heavy load conditions.

Several traffic and congestion control mechanisms are reviewed in the literatures ie [33], [34], [35], [38], [40], [42],[44] and [47]. In [33] the author has proposed the modified Forward Explicit Congestion Notification (FECN) adaptive congestion control scheme with rate control mechanism at cell level for the best effort class and guaranteed burst class. An output buffer management scheme is proposed in [34] for effective congestion control and to achieve high throughput.

The CAC and UPC congestion control schemes reviewed in [40] and showed that the CAC is an effective scheme in preventing congestion when all the sources comply, whereas UPC is effective in isolating sources which do not comply. Thus a CAC and UPC combined control schemes prevent complying traffic sources from being affected by the effect of non-complying traffic sources.

The author in [42] has reviewed 8 different congestion control methods. They are: 1.Selective re-transmission, 2.Leaky bucket, 3.Violation tagging, 4.Peak rate throttling, 5.Direct Policing, 6.Selective discarding, 7.Fairness queuing and 8.Fairness discarding. A comparison of all these methods has been done in terms of control motivation, control power, multiplexing gains and cost. The author concludes that the most successful congestion control method is that which provides sufficient motivation to users to implement the adoptive control and hence recommends the fairness discarding method for congestion control.

The ABR flow control mechanism based on the rate control scheme is used in ABR service to control the source traffic. It is an end-to-end flow control mechanism that carries the feedback in accordance with the changing ATM
network characteristics and thus controls the source rate. Resource Management (RM) Cells are special control cells, which carry feedback to the source. The source adopts the feedback and performs a dynamic traffic shaping. The ABR source then expects to experience a low cell loss ratio and obtain a fair share of the available bandwidth. The Usage Parameter Control (UPC) function generally referred as the ABR policer, which may be used by the network to enforce compliant behaviour from each ABR source.

Many authors [26],[27],[29],[30],[32],[35] and [39] have reviewed and analysed the congestion control schemes for ABR services particularly on rate based congestion control scheme. The credit and rate based congestion control schemes are reviewed and compared in [35] the conclusion is that the rate based scheme is preferred over the credit based scheme. FECN and BECN control schemes and PRCA control Algorithm are reviewed in detail for the ABR service. The rate based congestion control scheme is found to be the better approach for incorporating ABR traffic into ATM networks as the scheme is more flexible to design for different requirements of low cost, stability, efficiency etc. The author in [39] discusses the OSU scheme for congestion control in ABR traffic. This scheme for the simulation model has been adopted for the research in this thesis. The OSU scheme is a rate based congestion avoidance scheme that uses Explicit Rate (ER) indication mechanism for source rate control. The switches compute the load level and the ER value and ask the sources to adopt the same. The scheme achieves fairness and a good link utilisation.

During the establishment of an ABR connection a maximum bandwidth, known as the peak cell rate (PCR) and a minimum useable bandwidth, known as the minimum cell rate (MCR), are negotiated. Once the connection is established, the network will guarantee an agreed bandwidth equal to the MCR. The network reserves sufficient resources to ensure this, while also producing a low CLR. The network however provides a low CLR only to
those connections, which obey the agreed source behaviour. A policer is deployed to detect and isolate non-compliant connections, which do not obey the agreed source behaviour. Cell loss ratio accordingly is not guaranteed for such non-compliant connections. The network agrees to meet or exceed the negotiated QoS as long as the end-system complies with the negotiated traffic contract.

The ATM network avoids and controls congestion through the traffic management functions, the traffic and congestion control methods as described in previous paras. When the congestion control is considered for an ABR service, the traffic policer used to control the source traffic plays a key role in congestion avoidance. A detailed discussion of traffic policing and the need for its performance evaluation is in the next section.

2.6 Traffic Policing and its performance measurement in ATM networks

Traffic policing plays a vital role in the success of congestion avoidance schemes, which require the availability of an admission control scheme and a policing function. Admission control regulates the amount of guaranteed traffic accepted by the network. The policing functional monitors the connections to ensure that sources do not violate their agreed traffic rate specifications.

In admission control, the decision to accept a new call is based on the predicted network performance, taking into account the declared traffic characteristics. Users may deliberately or inadvertently exceed the traffic volume declared at call set-up and may, without a traffic policer jeopardise the congestion avoidance scheme. The ATM network without a traffic policer may encounter the following problems with violating sources.
1. The QoS particularly the CLR may increase significantly from the value negotiated with the user. This may happen to all the users sharing the common link due to a few violating sources.

2. The misbehaving user exceeding the usage of bandwidth to a large extent will automatically reduce the availability of bandwidth to the Co-users sharing a common link and denying them their fair share of bandwidth to an extent that may go below the guaranteed Minimum Cell Rate (MCR).

3. Many misbehaving users can push network nodes into congestion status, particularly on links with large propagation delays where the congestion feedback, to well-behaved users is delayed.

4. Statistical bandwidth gain is reduced without the policer.

In [20], it is shown that the link utilisation has increased from 50% to 80% with UPC/policer. The author in [19], has shown that the effective bandwidth increases with the policer and hence more sources can be accepted. It is shown that with a leaky bucket policer, the numbers of sources that can be admitted are 88 compared to 73 without the policer. The author has also shown that statistical multiplexing with a policing scheme improves.

The need for policing is more acute in a WAN environment rather than in a LAN environment [8]. In a LAN environment usually the local network administrator can control any misbehaving end stations or users through administrative means. That is, any uncontrolled usage of network resources can be detected, based on user complaints or routine checks. Once the misbehaving station is found it can be disconnected until the problem is solved. Deploying a policer in the above scenario may be more expensive. However, in a WAN environment due to the large number of connections sharing common links, the service provider has to ensure that well-behaved users are not seriously affected due to any particular misbehaving end station
Traffic and Congestion control in the ABR Service of ATM networks

or user. Such service guarantees can be provided by deploying a policer for each user at the input to the network.

Providing guarantee of QoS of IP over ATM needs policing. Many issues related to policing PCR of IP over CBR and VBR applications and the ACR of IP over ABR applications are presented in [21]. The author emphasises the minimum cell loss as the criteria for policing. The reason is that a single ATM cell loss would cause the entire IP packet loss and thus result in poor QoS.

A policer regulates the traffic from a source to be within the agreed rate. However, an inaccurate policer (as policing algorithm) may not maintain the traffic within the agreed limits. Actual policer accuracy [18] can be described in terms of deviation from an ideal policer (Refer fig. 2.6.1). Ideally the policer should detect a cell as non-conformed if a parameter is violated and as conformed if there is no violation. However, due to inherent errors in the algorithms the influence of delays will make the policer non-ideal and results in two types of error [18].

Type 1 errors occur when conformed cells are detected as non-conformed. Discarding the non-conformed cells will affect the cell-loss guarantee. The Type 2 errors occur when non-conformed cells are not detected. Allowing non-conformed cells into the network will load the buffer with excess cells and affect the QoS within the network. If the network nodes do not reserve extra buffer space, they may become congested.

Performance measurement to determine the magnitude of these errors is needed to avoid the impact that these errors will have on the user and the network. Suitable corrective actions such as including maximum delay bounds in the police algorithm to avoid Type 1 errors, and reserving extra resources to compensate for Type 2 error, can be taken. If an inaccurate policer allows cells at higher than the agreed rates, then the network will not maintain its QoS guarantees. Similarly, well-behaved users may suffer cell-
loss if the policer wrongly detects them as exceeding the cell rate. The accuracy and the performance of a policer depends on the Algorithm used and the magnitude of variations in the network parameters, particularly the Round Trip Delay (RTD).

Evaluation of the performance of policing mechanisms, in terms of number of violating cells being detected appears in [17]. The author has compared the performance of the leaky bucket, jumping window and EWMA policing schemes and found that the leaky bucket policer is the best, followed by Jumping window and EWMA. The author has found that the EWMA scheme has detected 3.78% of the cells wrongly as rate violating whereas this figure is 2.8% in the jumping window scheme.

The author in [19] has evaluated the performance of a policing scheme through the measurement of statistical multiplexing gain by introducing the concept of effective bandwidth, as outlined below.

In [19] an appropriate measure of effectiveness of a policing scheme is given by

\[ G_i = V_i^p - V_i^l / V_i^p - V_i^0 \]

For a source class i, \( G_i \) denotes the statistical multiplexing gain, \( V_i^0 \) denotes the effective bandwidth assuming that all sources are well-behaved, \( V_i^p \) the bandwidth allocated the source in the peak bandwidth assignment scheme and \( V_i^l \) denotes the effective bandwidth under certain number of badly behaved sources. Different policing schemes give different values to \( V_i^l \).

For an ideal scheme, \( G_i = 1 \) and \( V_i^l = V_i^0 \) generally \( 0 \leq G_i \leq 1 \)

The value of \( G_i \) for different policing mechanism under similar conditions will determine the performance of each policer. The Author has measured and compared the performances between a single and multiple leaky bucket policing mechanisms. It is concluded that the statistical multiplexing with
Traffic and Congestion control in the ABR Service of ATM networks

multiple leaky bucket policing mechanism does improve the multiplexer efficiency and has the advantage in accepting 50% more number of sources over single leaky bucket policing mechanism.

Many literatures and journals have published on the performance of policer using various methods of leaky bucket mechanism. The selected ones are [23],[31],[37],[39],[41] and [46]. The mechanism involves in policing peak cell rate (PCR) and sustainable cell rate (SCR). The performance is measured in terms of cell loss to the well-behaved users. The author of [39] emphasises the statistical gain obtained by introducing the policing. The performance analysis of 4 different UPC algorithms has been discussed in [41]. The algorithms are 1. Triggered Jumping Window (TJW), 2. Sliding Window (SW), 3. Exponentially Weighted Moving Average (EWMA) and 4. Leaky Bucket (LB). The performance is measured in terms of quantity of tagging excess cells, speed of reaction and cell delay. The author concludes that TJW algorithm is the best compared out of the other 3 algorithms.

It can be seen from the above examples that a performance measurement of policing mechanisms is needed to determine their accuracy in terms of cell-loss, bandwidth reduction or statistical bandwidth gain. Such measurements are needed to find the effectiveness of a policing mechanism to ensure the QoS being agreed with the user. All the policers reviewed so far, are for the ATM service categories that are policed at peak cell rate (PCR). However, in an ABR service, the source should not exceed the allowed cell rate (ACR), which varies dynamically during the lifetime of the connection. The value of ACR is bounded by the Explicit Rate (ER) feedback sent by the network to the source. Hence the policer has to ensure that the source does not exceed the ER. Because of this, the policer has to dynamically adopt a new police rate every time the network sends a new ER feedback to the source. A review of the performance measurement of such a policer appears in the next section.
2.7 The ABR Police and its performance measurement

The ABR policer, as explained in the previous section, monitors the source traffic with respect to a police rate that changes dynamically. The policing mechanism uses two types of algorithms. The first type of algorithm (ex; Leaky bucket) is used to check the conformance of a cell with respect to the police rate. The second type of algorithm (ex: N_Store Algorithm B) evaluates a new police rate every time it receives an Explicit Rate (ER) feedback from the network. Since it takes one round trip delay for the ER feedback to reach the source and cells from the source to arrive at policing point after adopting the new feedback rate, the policer has to wait for a one-trip delay before enforcing this rate on the incoming cells. Hence the second type of algorithm determines the timetable for enforcing the respective policing rates.

The performance evaluation of the ABR policer hence includes the accuracy with which the police rate and schedule are determined, along with the accuracy of conformance check. Thus measuring the performance of ABR policer is different from that in the non-ABR services.

In the previous section several papers were reviewed that measure the accuracy and performance of the policer. The measurement of accuracy, in terms of bandwidth saving and the number of police rates generated over a period has been performed [11] to compare the two police rate algorithms N_Store and 2_Store. The author in [22] has investigated 3 policing mechanisms in terms of their effectiveness in ensuring the fairness among competing ABR users. The 3mechanisms are Algorithm A, B (as in [1]) and C (a new policing scheme proposed by the author). For violating sources the author has measured, a) performance of Algorithm A and B in terms of percentage of non conforming cells detected by the policer and b) performance of Algorithms A, B and C for throughput. However, no
literature has been found on the performance of the ABR policer in terms of Type1 and Type2 errors. This thesis addresses this gap.

The main topics of the thesis are:

a) Introduction of a new ABR policer performance measurement method.

b) Measurement of the magnitude (through simulation) of Type1 and Type2 errors in terms of cell loss and excess-cells for the ABR policer with 3 different police rate algorithms.

c) Analysis of the results and detection of the errors in the algorithm.

d) Improvement and simplification of the algorithm.

e) Comparison of the results of all the 4 Algorithms (Algorithm A, 2_Store, N_Store and simplified N_Store_I).

A detailed study of the ABR policing algorithms is needed before proceeding with the performance analysis. Chapter 3 describes all the algorithms used in policing along with the ABR flow control, providing the necessary technical background for the performance analysis.
3. An overview of the ABR policer

3.1 Introduction

This chapter describes the basic principle on which the ABR policer works and the different algorithms used to implement it which provides a complete technical background for the policer analysis in chapter 4. Section 3.2 describes the traffic contract between the user and the network service provider. The cell conformance definition, which forms the basic principle for the policing function, is also described. The policer performs the conformance check of an incoming cell with respect to a reference rate. TM 4.0 has specified the DGCRA algorithm for the cell conformance check. Algorithms A and B evaluate the police reference rate dynamically, based on three factors; i) the feedback rate provided by the RM cell, ii) the time at which the RM cell crosses the policing point and iii) the parameters $\tau_2$ (Maximum round trip delay) and $\tau_3$ (Minimum round trip delay) set by the network operator. Operation of each algorithm is explained in sections 3.4.1 - 3.4.4.

The measurement of the performance of the policer for each of the above police rate algorithms appears in chapter 4.
3.2 Traffic contract and Cell Conformance

The network, as a part of traffic management, monitors and regulates the traffic flow from its source through the UPC at the UNI. Network operators may use the UPC function as a policer. Before deploying the policer both the ABR user and the network should enter into a contract on a set of agreed i) traffic parameters, ii) source behaviour, iii) cell conformance definition and iv) compliant connection definition.

A set of negotiated traffic and quality parameters specify the traffic contract of a connection. The traffic characteristics are specified in terms of a set of source traffic parameters such as PCR, MCR, ACR and CDVT and a set of QoS parameters such as CLR and conformance definition, for each direction of the connection. The contract is established when the end-systems agree to these traffic parameters and CDVT during the call setup.

The above traffic parameters are used by CAC to allocate resources and also to provide required parameter values for the policing function, to meet the network performance objectives. The cell conformance testing is done at the UNI by the policer, by taking the relevant information from the above parameters. The conformance definition in the ABR service refers to the specified source behaviour. The cell conformance of a connection is defined in terms of traffic parameters, while the conformance algorithm is designed based on the conformance definition. The conformance definition is specified in the traffic contract. This is discussed in section 3.2.2.

3.2.1 Cell Conformance and Connection Compliance

The cell conformance check is performed on all forward data cells, as they pass through a policer (located at UNI). The algorithm starts at the arrival of first cell and then continuously checks subsequent cells as being either conforming or non-conforming. The connection may be considered
An Overview of the ABR policer

compliant if all the cells on it are conforming. However, even under ideal conditions not all the cells may be conforming due to tolerance errors. Hence a connection is considered as compliant if the total non-conforming cells fall below a set threshold. The network operator sets a threshold for the total non-conformed cells over a particular period. If the total non-conformed cells over a connection exceed the threshold limit then it is considered as a non-compliant connection. The network may not commit QoS to such connections.

3.2.2 Cell conformance definition

The cell conformance definition refers to the ABR source behaviour specified in the contract. In the ABR service, the source traffic characteristic changes dynamically, due to the adaptation of feedback provided by the network. A Generic cell rate algorithm (GCRA) [1] is used to define the conformance for Variable bit rate (VBR), Constant bit rate (CBR) and Unavailable bit rate (UBR) services. The GCRA provides the Peak cell rate (PCR) conformance cell by cell by comparing the inter-cell interval with respect to a reference interval 1/PCR. In the ABR service, a Dynamic GCRA (DGCRA) provides the Allowed cell rate (ACR) conformance, cell by cell, by comparing the inter-cell interval with respect to a reference interval 1/PACR (i.e.,1/Potential ACR). The ABR conformance definition depends on two parameters that vary throughout the connection time. They are 1) the feedback rate delivered by the RM cell and 2) the Round Trip Delay (RTD). Basically, the incoming cell at any point of time ‘T’ has to conform with the feedback rate sent across the User network interface (UNI) at a time RTD ago ie T-RTD.
3.2.2.1 Round Trip Delay (RTD)

The round trip delay is the sum of several delays incurred along the path from the policing point to the source and back. One or more components involved in this delay are; 1. propagation delay due to physical media, 2. transmission delay induced by the transmitting End systems, 3. Switch processing delay, 4. buffering and scheduling delay that occurs at the switch node and 5. queuing delay at a switch node.

The propagation, transmission and switch processing delays are generally fixed. The buffering and scheduling delays are variable and the total delay is referred to as Cell Delay Variation (CDV) [1]. The upper bound on CDV is referred as CDV Tolerance (CDVT) which is specified before establishing the connection. CDVT is also referred as $\tau_1$.

The queuing delay in the ABR service is unpredictable and varies over a long range and changes rapidly [15] particularly if the ABR connections are implemented at a lower priority. The VBR and CBR traffic that share a common link with ABR traffic are given the higher priority. The low priority ABR traffic at times shares a small portion of the bandwidth. Hence, the ABR cells may incur long delays. In the absence of VBR or CBR traffic, the delay variation on the ABR cells will be very minimal. Thus the ABR cell delay, which depends on the VBR and CBR traffic loads, causes the round trip delay to fluctuate throughout the connection.

Feedback generated at a network interface may reach the source after a delay which is either equal to a fixed delay or equal to (fixed delay + queuing delay), depending upon the absence or presence of CBR and VBR traffic. The cells transmitted from the source may encounter similar delays before reaching the policing point in the network interface. The backward delay may not be equal to the forward delay particularly when there are several sub networks (network segments) between the source and the network Interface. Thus the RTD between the source and the policing (network interface) point
is bound by two delays, $\tau_2$ and $\tau_3$ \cite{14}, \cite{15}. The $\tau_2$ and $\tau_3$ are upper and lower bound on the delay after which a feedback rate carried by backward RM cells (across network interface / policing point in backward direction) is expected to be reflected (through adopted source rate) at the same policing point in the forward direction \cite{1}.

3.2.2.2 The ABR conformance definition

ABR conformance is defined \cite{1} as follows.

All CLP=0 forward cells received from a source are said to conform,

- If the inter-cell interval of the incoming cell is greater than $(1/MCR- \tau_1)$, where $\tau_1$ is CDVT.

- The inter-cell interval between the cell arrived at the policer after adopting the feedback rate and the cell arrived previous to that,

1) Should account for Explicit Rate (ER) feedback that is conveyed through backward RM-cell that is transmitted at $\tau_2$ or latter time ago across the interface on the backward connection.

2) Should not account for feedback (ER) that is conveyed in backward RM-cell that is transmitted across the interface on the backward connection earlier than $\tau_3$ or earlier time ago.

The DGCRA algorithm used for the conformance of cells of rate $1/ER = 1$ and $CDVT = \tau_1$ is denoted by DGCRA $(I(k), \tau_1)$. $I(k)$ is determined based on the ERs received between $\tau_3$ and $\tau_2$ times ago from the time of the arrival of a forward cell from the source as per the above definitions 1) and 2).

The ABR policer may adopt any algorithm that meets the above conformance requirement. The different algorithms specified by ATM Forum in its TM specification \cite{1} are discussed in the next section.
3.3 The ABR Policing related algorithms

The conformance check by the policer must conform to the definition given in the previous section. The ATM Forum has specified a DGCRA algorithm to check the conformance of an incoming cell with respect to a reference rate (police rate). It has specified two algorithms, Algorithm A and B (2_Store), to determine the police rate. The N_Store algorithm as an improvement over the 2_Store algorithm is proposed through ATM Forum contributions. We outline and show two serious errors in the N_Store algorithm. The DGCRA algorithm needs two parameters CDVT and Police rate for a conformance check whereas the other two algorithms need 4 parameters to evaluate the police rate and its schedule. They are: 1) arrival times of RM cells that crossed the UNI, 2) feedback Explicit Rates (ER) carried by RM cells, 3) the upper bound delay $\tau_2$ and 4) the lower bound delay $\tau_3$.

For all the algorithms to operate, an estimation of above parameters is needed. The CDVT estimation is achieved during the call setup and its value will be available. Arrival times of RM cells and the feedback rates that they carry are copied and stored by the algorithms (A or B) before sending them to the source across UNI. The major problem lies in the estimation of $\tau_2$ and $\tau_3$. Unlike CDVT, the $\tau_2$ and $\tau_3$ parameters are very difficult to estimate for the following reasons [14].

- Since the RTD varies dynamically, it is therefore difficult to estimate an accurate value for it. This is particularly so in segmented networks.

- Even though the RTD may be estimated accurately, the network has to determine $\tau_2$ and $\tau_3$, based on the estimation that penalises malicious users while being lenient on well-behaved users (ie where well-behaved users are not penalised due to unpredictable RTD).

The values of $\tau_2$ and $\tau_3$ for a connection are chosen so that the required QoS of the connection is met without affecting the QoS of the existing connections.
Because of these constraints, the determination of $\tau_2$ and $\tau_3$ is left to the network operator, who will determine the values based on the experience, specific connection type and its propagation delay. In fact $\tau_3$ may be set to propagation delay. It is only $\tau_2$ that need to be estimated.

Once the parameters $\tau_1$, $\tau_2$ and $\tau_3$ used by the ABR Conformance definition are determined, they may be specified explicitly or negotiated by the network at the time of call establishment.

The DGCRA algorithm is described in the next section. Algorithms A and B are explained in detail in the following sections, 3.4.1 to 3.4.3

### 3.3.1 Dynamic Generic Cell Rate Algorithm (DGCRA)

DGCRA is used in the ABR service to specify the conformance to Explicit Rate (ER) on a connection (VC OR VPC) at the public or private UNI. The conformance is on cell by cell basis. The DGCRA is an extension of GCRA.

#### 3.3.1.1 DGCRA using leaky bucket algorithm

![Generic Cell Rate Algorithm (GCRA)](Fig3.3.1.1: Generic Cell Rate Algorithm (GCRA))
The GCRA algorithm (refer to fig. 3.3.1.1) used to find the conformance of cells with a rate PCR is represented as GCRA(I,τ₁). Where I is the peak emission interval, I = 1/PCR and τ₁ is the Cell delay variation tolerance (CDVT). The CDVT is the upper bound of the CDV introduced by the ATM layer functions, multiplexing stage up to measuring point. τ₁ characterises the maximum limit on the cell delay. The conformance of an incoming k\textsuperscript{th} cell arriving at a time a(k) is checked as follows.

If \( a(k) \geq T(k) - \tau_1 \) then the cell is conformed otherwise it is non-conformed. T(k) is the theoretical arrival time evaluated by the algorithm as follows.

\[
\begin{align*}
\text{For } k \geq 2 & \quad \text{[For } k=1, \text{ } T(1)=a(1)] \\
T(k+1) &= T(k) + I \quad \text{if } \tau_1 \geq [T(k) - a(k)] \geq 0 \quad \text{Cell is conformed} \\
&= a(k) + I \quad \text{if } [T(k) - a(k)] < 0 \quad \text{Cell is conformed} \\
&= T(k) \quad \text{if } [T(k) - a(k)] > \tau_1 \quad \text{Cell is non-conformed}
\end{align*}
\]

ie, if the cell arrives earlier than the scheduled time T(k) but within the CDVT tolerance.

ie, if the cell arrives after the scheduled time T(k)

ie, if the cell arrives earlier than the scheduled time T(k) but out of the CDVT tolerance

Unlike GCRA, the increment I in DGCRA change dynamically with time as determined by the ABR feedback. DGCRA checks conformance only for CLP = 0 cells. This is particularly mentioned because many times RM cells
are sent out of rate with CLP=1. When a $k^{th}$ CLP=0 cell arrives, the DGCRA first calculates the increment $I(k)$ and then checks the conformance of the cell. The DGCRA algorithm used for the conformance of cells of rate $1/ER = I$ and CDVT $\tau_1$ is denoted by DGCRA $(I(k), \tau_1)$.

3.3.1.2 DGCRA using virtual scheduling algorithm:

For $k=1$: $LVST = t(1), I(old) = I(1)$

At the arrival of $k^{th}$ cell at $t(k)$ for $k \geq 2$, the DGCRA calculates $I(k)$ and then checks the conformance of the cell.

* If $a(k) > LVST + \text{Min}[I(k), I(old)] - \tau_1$ then the cell is conforming and the algorithm will be updated for $LVST$ and $I(old)$ as given below, otherwise the cell is non-conforming and the algorithm will not be updated.

$a(k)$ is the actual arrival time, $LVST$ is the last virtual schedule time.

The actual arrival time is checked against the $[\text{Scheduled arrival time} - \text{CDVT}]$

Scheduled arrival time $= LVST + \text{Min of inter-cell interval of new and old rates}$

If the cell arrives earlier than the scheduled arrival time by a factor 0 to $\tau_1$, then the $LVST$ is taken as a minimum of $I(k)$ and $I(old)$. If the cell arrives at a time equal or after the scheduled arrival time, then $LVST$ is taken as $a(k)$.

Hence,

- Set $LVST = \text{Max}[t(k), LVST + \text{Min}(I(k), I(old))]$
- Set $I(old) = I(k)$

If the cell arrives at a time before the scheduled arrival time, then the cell is not conforming and the algorithm will not be updated.
3.3.1.3 Determination of police rate I(k) and its schedule

I(k) is the inverse of the police rate, the determination of which is algorithm specific. The police rate is generally referred to as Potential ACR (PACR). I(k) = 1/PACR. The schedule required to implement I(k) depends on two additional delay parameters, \( \tau_2 \) and \( \tau_3 \), for the connection. The sequence \( \{I(k), k \geq 1\} \) of increments, which are successively used at arrival times \( \{t_a(k), k \geq 1\} \) CLP=0 cells at the interface, depends on the sequence of feedback explicit rates \( \{ER(j), j \geq 1\} \) sent across the interface at times \( \{t_b(j), j \geq 1\} \) through backward RM cells. These sequence of ERs are mapped on to I(k) as follows.

- The increase in ER is not mapped onto I(k) until after a lag of \( \tau_3 \)
- The decrease in ER is not mapped on to I(k) until after a lag of \( \tau_2 \).

That means any increase in feedback rates is enforced after an estimated lowest round trip delay and a decrease in rate after the longest round trip delay. With this definition, the policer will be monitoring the cells at higher rates for a longer period. This is done mainly to avoid cell-loss to the well-behaved user due to the policer.

The police rate and its schedule are determined as outlined in the conformance definition by Algorithm A, B (2_Store) and N_Store are given in section 3.3.2 to 3.3.4. A graphical representation of operations of the 3 algorithms is given in fig 3.3.2, 3.3.3 and 3.3.4. The evaluated police rates and their schedules are given in tables 3.3.2, 3.3.3 and 3.3.4. A comparative table is given in table 3.3. The four main parameters required by the algorithm to derive the Police Rate (PR) and its schedule are Explicit Rate (ER), the arrival time of RM cell (at UNI), maximum bound delay \( \tau_2 \) and Minimum bound delay \( \tau_3 \). The figs. 3.3.2 to 3.3.4 show the arrival of 14 Explicit rates ER1 to ER14 at times \( t_{ER1} \) to \( t_{ER14} \) (not shown in the fig.) within a period of 13milli sec. \( \tau_2 \) and \( \tau_3 \) are chosen to be 5ms and 1ms respectively. PR1 to PR14 and \( t_{PR1} \) to \( t_{PR14} \) (not shown in the fig.) are the
corresponding police rates and schedules derived by the algorithms. Police rates PR1 to PR14 will have the same values of ER1 to ER14 respectively. ER0 and PR0 are the initial rates. The police rate envelope produced by each algorithm is described below.

### 3.3.2 Algorithm A

At the arrival of the \(k\)th cell at \(a(k)\) at the interface, the algorithm evaluates the new police rate \(\text{PACR}_k\) as follows.

It first evaluates \(\text{ER}_{\text{max}}\) which is the maximum of the ER values (feedback rates) received from the backward RM cells within the interval \([a(k)-\tau_2, a(k)-\tau_3]\). Then it chooses the value \(\text{ER}1\) which is the last ER conveyed by a backward RM cell previous to time \(a(k) - \tau_2\). The new police rate \(\text{PACR}_k\) will be, \(\text{PACR}_k = \text{Max} [\text{ER}1, \text{ER}_{\text{max}}]\). The evaluated rate must be within PCR and MCR.

Hence \(\text{PACR}_k = \text{Min} [\text{PCR}, \text{Max} (\text{PACR}_k, \text{MCR})]\)

Finally the increment \(\text{I}(k) = 1/\text{PACR}_k\)

Refer to fig 3.3.2 for police rate envelope and table 3.3.2 and 3.3.2a for police rates and the schedules obtained from Algorithm A. The police rate is determined at the arrival time \(t_a\) of a forward cell. The police rates PR1, PR2, PR3, PR13 and PR14 are chosen after a time delay of \(\tau_3\) from the arrival times of respective feedback rates ER1, ER2, ER3, ER13 and ER14. The police rate PR6 that is the maximum of ER4 to ER10 is chosen after a delay, \(\tau_2\), from the arrival time of ER4. Similarly PR7 which is the maximum of ER7 to ER12 is chosen after a delay \(\tau_2\) from ER7.

<table>
<thead>
<tr>
<th>ER</th>
<th>New-Police rate</th>
<th>New-Police rate Schedule</th>
<th>Delivered-Police rate</th>
<th>Delivered-Police rate Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0</td>
<td>tER0</td>
<td></td>
<td>PR0</td>
<td>tER0</td>
</tr>
<tr>
<td>ER1</td>
<td>PR1</td>
<td>tER1 + (\tau_3)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### An Overview of the ABR policer

Table 3.3.2: Police rates and their schedules evaluated by Algorithm A

<table>
<thead>
<tr>
<th>ER2</th>
<th>PR2</th>
<th>tER2 + τ₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>ER3</td>
<td>PR3</td>
<td>tER3 + τ₂</td>
</tr>
<tr>
<td>ER4</td>
<td>PR4</td>
<td>tER4 + τ₂</td>
</tr>
</tbody>
</table>

The real time crosses tER1 + τ₂ and the chosen police rate is PR1 which is the maximum of PR1 and PR0 scheduled within [Real time - (τ₂ - τ₁)]

| PR1  | tER1 + τ₂ |

The real time crosses tER2 + τ₂ and the chosen police rate is PR2 which is the maximum of PR0 to PR2 scheduled within [Real time - (τ₂ - τ₁)]

| PR2  | tER2 + τ₂ |

The real time crosses tER3 + τ₂ and the chosen police rate is PR3 which is the maximum of PR0 to PR3 scheduled within [Real time - (τ₂ - τ₃)]

| PR3  | tER3 + τ₃ |

| ER5  | PR5  | tER5 + τ₂ |

| PR6  | tER6 + τ₂ |
|------|------|-----------|
| ER7  | PR7  | tER7 + τ₂ |
| ER8  | PR8  | tER8 + τ₂ |
| ER9  | PR9  | tER9 + τ₂ |
| ER10 | PR10 | tER10 + τ₂ |

The real time crosses tER4 + τ₂ and the chosen police rate is PR6 which is the maximum of PR4 to PR10 scheduled within [Real time - (τ₂ - τ₃)]

| PR6  | tER4 + τ₂ |

The real time crosses tER5 + τ₂ and the chosen police rate is PR6 which is the maximum of PR5 to PR10 scheduled within [Real time - (τ₂ - τ₃)]

| PR6  | tER5 + τ₂ |

| ER11 | PR11 | tER11 + τ₂ |

The real time crosses tER6 + τ₂ and the chosen police rate is PR6 which is the maximum of PR6 to PR11 scheduled within [Real time - (τ₂ - τ₃)]

| PR6  | tER6 + τ₂ |

| ER12 | PR12 | tER12 + τ₂ |

The real time crosses tER7 + τ₂ and the chosen police rate is PR7 which is the maximum of PR7 to PR12 scheduled within [Real time - (τ₂ - τ₃)]

| PR7  | tER7 + τ₂ |

| ER13 | PR13 | tER13 + τ₃ |

The real time crosses tER8 + τ₂ and the chosen police rate is PR10 which is the maximum of PR8 to PR12 scheduled within [Real time - (τ₂ - τ₃)]

| PR10 | tER8 + τ₂ |

The real time crosses tER13 + τ₂ and the chosen police rate is PR13 which is the maximum of PR9 to PR13 scheduled within [Real time - (τ₂ - τ₃)]

| PR13 | tER13 |

| ER14 | PR14 | tER14 + τ₃ |

The real time crosses tER14 + τ₃ and the chosen police rate is PR14 which is the maximum of PR9 to PR13 scheduled within [Real time - (τ₂ - τ₃)]

| PR14 | tER14 + τ₃ |
### Algorithm A

<table>
<thead>
<tr>
<th>Police rate</th>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0</td>
<td>tER0</td>
</tr>
<tr>
<td>PR1</td>
<td>tER1 + τ₁</td>
</tr>
<tr>
<td>PR2</td>
<td>tER2 + τ₁</td>
</tr>
<tr>
<td>PR3</td>
<td>tER3 + τ₁</td>
</tr>
<tr>
<td>PR6</td>
<td>tER4 + τ₂</td>
</tr>
<tr>
<td>PR7</td>
<td>tER7 + τ₂</td>
</tr>
<tr>
<td>PR10</td>
<td>tER8 + τ₂</td>
</tr>
<tr>
<td>PR13</td>
<td>tER13 + τ₃</td>
</tr>
<tr>
<td>PR14</td>
<td>tER14 + τ₃</td>
</tr>
</tbody>
</table>

Table 3.3.2a: List Police rates and their schedules delivered by Algorithm A

---

**POLICE RATE ENVELOPE**

(T₁=5ms and T₂=1ms)

![POLICE RATE ENVELOPE](image)

**Fig. 3.3.2**

Police rate envelope - Algorithm A
The algorithm A is a perfect model that checks the conformance exactly as the definition requires and memorises all the feedback rates conveyed by RM cells. It is very complex to implement.

At the arrival time \( t \) of every forward cell, it has to compute \( ER_1 \) and \( ER_{\text{max}} \) for which it has to maintain the storage of all Explicit Rates received within a time \( (t - \tau_2, t) \) and the last RM cell received previous to \( (t - \tau_2) \). Even though the algorithm memorises all the rates, it is rather complex to implement. Hence the ATM Forum has specified a simpler Algorithm B which is described below.

### 3.3.3 Algorithm B

Like Algorithm A, Algorithm B derives the PACR from the explicit rates \( ER(j) \) received in the backward RM-cells. A received backward RM-cell results in a delayed increase or a delayed decrease of the police rate PACR. Rate increases are scheduled with a delay of \( \tau_3 \) or earlier. Rate decreases are scheduled with a delay of \( \tau_2 \) or later.

Algorithm B is a simple algorithm consisting of two sets of stores for storing at most 2 explicit rate changes and their corresponding schedules. The first set of rates and schedule are stored respectively in DER\(_{\text{first}}\) and \( t_{\text{first}} \) for any increase in rates, whereas the second set of rate and schedule are stored respectively in DER\(_{\text{last}}\) and \( t_{\text{last}} \) for a decrease in rate. An increase or decrease in explicit rate is determined by comparing \( ER(j) \) with DER\(_{\text{first}}\) and with the PACR the current police rate. The DER\(_{\text{first}}\) will come into effect when the real time crosses \( t_{\text{first}} \).

Initially, the first feedback explicit rate (received by the backward RM cell) is stored in DER\(_{\text{first}}\) and the corresponding schedule in \( t_{\text{first}} \). If the subsequent feedback explicit rate arrives before the schedule \( t_{\text{first}} \) and is an increase over DER\(_{\text{first}}\) then it replaces the DER\(_{\text{first}}\). If it is a decrease in
rate then the rate and its schedule replace DER_last at t_last respectively. The way the algorithm store increases and decreases in rates under various conditions are given below [1], [7]. The TM specification [1] provides the pseudo code.

**Rate increase:**

If ER(j) is greater or equal to DER_first, then a rate increase is scheduled. Both DER_first and DER_last are replaced by ER(j). That means any increase in rate erases previous rate increases. The t_last is unscheduled ie set to zero. The t-first is evaluated as follows:

a. The t_first is not changed (ie retains previous value of t_first) if ER(j) is less than PACR.

b. The t_first is set to minimum of t_first and tb(j) + τ3 if ER(j) is greater than PACR.

c. If t_first is zero then it is overwritten by tb(j) + τ3.

**Rate decrease:**

1. If ER(j) < DER_first, then a rate decrease is scheduled. DER_last and t_last are overwritten by ER(j) and tb(j) + τ3 respectively.

**Rate delivery:**

When time reaches t_first, DER_first is transferred to police rate PACR and DER_last and t_last are transferred to DER_first and t_first respectively.
An Overview of the ABR policer

### Algorithm B (2 Store)

<table>
<thead>
<tr>
<th>ER</th>
<th>Police rate DER_first</th>
<th>Schedule t_first</th>
<th>Police rate DER_last</th>
<th>Schedule t_last</th>
<th>Delivered Police rate and schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0</td>
<td>0</td>
<td></td>
<td>PR0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ER1</td>
<td>PR1</td>
<td>tER1 + t3</td>
<td>PR1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ER2</td>
<td>PR2</td>
<td>tER1 + t3</td>
<td>PR2</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ER3</td>
<td>PR3</td>
<td>tER1 + t3</td>
<td>PR3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ER4</td>
<td>PR3</td>
<td>tER1 + t3</td>
<td>PR4</td>
<td>tER4 + t2</td>
<td></td>
</tr>
</tbody>
</table>

The real time crosses $tER1 + t3$ and PR3 stored in DER_first is delivered for policing. The PR4 and $tER4 + t2$ stored in DER_last and t_last are copied to DER_first and t_first respectively.

<table>
<thead>
<tr>
<th>PR4</th>
<th>tER4 + t2</th>
<th>PR4</th>
<th>tER4 + t2</th>
<th>PR3</th>
<th>tER1 + t3</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ER5</th>
<th>PR4</th>
<th>tER4 + t2</th>
<th>PR5</th>
<th>tER5 + t2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ER6</td>
<td>PR6</td>
<td>tER4 + t2</td>
<td>PR6</td>
<td>0</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>ER7</td>
<td>PR6</td>
<td>tER4 + t2</td>
<td>PR7</td>
<td>tER7 + t2</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>ER8</td>
<td>PR6</td>
<td>tER4 + t2</td>
<td>PR8</td>
<td>tER8 + t2</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>ER9</td>
<td>PR6</td>
<td>tER4 + t2</td>
<td>PR9</td>
<td>tER9 + t2</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>ER10</td>
<td>PR6</td>
<td>tER4 + t2</td>
<td>PR10</td>
<td>tER10 + t2</td>
</tr>
</tbody>
</table>

The real time crosses $tER4 + t2$ and PR6 stored in DER_first is delivered for policing. The PR10 and $tER10 + t2$ stored in DER_last and t_last are copied to DER_first and t_first respectively.

<table>
<thead>
<tr>
<th>PR10</th>
<th>tER10 + t2</th>
<th>PR10</th>
<th>tER10 + t2</th>
<th>PR6</th>
<th>tER4 + t2</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ER11</th>
<th>PR10</th>
<th>tER10 + t2</th>
<th>PR11</th>
<th>tER11 + t2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ER12</td>
<td>PR10</td>
<td>tER10 + t2</td>
<td>PR12</td>
<td>tER12 + t2</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>------------</td>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td>ER13</td>
<td>PR13</td>
<td>tER10 + t2</td>
<td>PR13</td>
<td>0</td>
</tr>
</tbody>
</table>

PR13 is not scheduled at $tER13 + t3$ because $tER10 + t2 < tER13 + t3$.

The real time crosses $tER10 + t2$ and PR13 stored in DER_first is delivered for policing. The PR13 and 0 stored in DER_last and t_last are copied to DER_first and t_first respectively.

<table>
<thead>
<tr>
<th>PR13</th>
<th>0</th>
<th>PR13</th>
<th>0</th>
<th>PR13</th>
<th>tER10 + t2</th>
</tr>
</thead>
</table>

| ER14| PR14      | tER14 + t3 | PR14      | 0          |

The real time crosses $tER14 + t3$ and PR14 stored in DER_first is delivered for policing. The PR14 and 0 stored in DER_last and t_last are copied to DER_first and t_first respectively.

| PR14| 0          | PR14| 0          | PR14| tER14 + t3 |

**Table 3.3.3: Police rates and their schedules evaluated by Algorithm B (2_Store)**
An Overview of the ABR policer

<table>
<thead>
<tr>
<th>Algorithm B (2 Store)</th>
<th>Police rate</th>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>PR3</td>
<td>tER1 + τ3</td>
<td></td>
</tr>
<tr>
<td>PR6</td>
<td>tER4 + τ3</td>
<td></td>
</tr>
<tr>
<td>PR13</td>
<td>tER10 + τ2</td>
<td></td>
</tr>
<tr>
<td>PR14</td>
<td>tER14 + τ3</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.3.3a: List Police rates and their schedules delivered by Algorithm B (2_Store)

Fig. 3.3.3 Police rate envelope - Algorithm B (2_Store)
Refer to fig 3.3.3, 3.3.3a and table 3.3.3 for police rates and the schedules obtained from Algorithm B. Initially DER_first and the current police rate PACR in force are set to PR0 with schedule t_first being set to 0. At the arrival of the first feedback explicit rate ER1 which is an increase over PACR and DER_first forces PR1 to replace PR0 and schedules it after τ3 ie tER1 + τ3 where tER1 is the time of arrival of ER1. The DER_last is also replaced by PR1 but with schedule set to zero. Further increases in rates over PACR and DER_first overwrite PR1 but keeps the earliest schedule (tER1 + τ3).

PR4, a decrease in rate is stored in DER_last and scheduled after t2. PR3, at its scheduled time (tER1 + τ3), is moved from DER_first and delivered as police rate to PACR and then the PR4 and tER4 + τ2 are moved from DER_last and t_last to DER_first and t_first respectively as shown in the table 3.3.3.

PR5, which is a decrease over PR4, is stored in DER_last. PR6 is an increase in rate, which replaces PR4 in DER_first without changing the schedule time in order to keep the earliest schedule. Hence PR6 is delivered at PR4 schedule. PR7 to PR10 are decrease in rates over PR6 hence stored in DER_last. After DER_last is overwritten by PR7 to PR10 in order, DER_last and t_last are left finally with PR10 and its schedule (tER10 + τ2). After DER_first delivering PR6, the PR10 is moved from DER_last to DER_first.

At the arrivals of decrease in rates ER11, ER12 the corresponding police rates PR11 is stored in DER_last, which is overwritten by PR12. At the arrival of increase in rate ER13, the corresponding police rate PR13 replaces PR10 stored in DER_first and ER11 and ER12 stored in DER_last. The schedules in t_first and t_last will be tER10 + τ3 and zero respectively. Once the real time crosses t_first, the PR13 is delivered at PR10 schedule time and PR13 from DER_last is moved to DER_first without any schedule. Finally PR14 being an increase over the PR13 replaces PR13 and is scheduled after τ3 (tER14 + τ3). The 2_Store algorithm leaves an envelope of police rate as shown in table 3.3.3a and by the dotted lines in fig. 3.3.3.
3.3.4 Cyclic N_Store Algorithm

The major inaccuracy in the 2_Store policer is its inability to store more than two rates. A large number of stores will reduce this inaccuracy. The N_Store policer [10], [2], which is an extension of 2_Store policer, has the following important properties:

N = 2 is identical to the 2-store policer.

For large values of N, as it memorises practically all the rates and times of arrival, the algorithm tends to be an ideal policer similar to Algorithm A. However the errors in the algorithm allow for the improved performance of the algorithm over 2_Store and avoids the algorithm in achieving a performance equivalent to Algorithm A.

Cyclic N_Store algorithm operates on the same principle of 2_Store algorithm except that the choice of 2 stores now cycles over N stores. That is, whenever the algorithm receives an explicit rate through a backward RM cell, it stores the new rate and its schedule in the incremented next store, which cycles after the Nth store. The N is incremented after a minimum period of \( \tau_3/(n-1) \), where \( \tau_3 \) is the minimum round-trip delay between the policer and the source.

Basically the Cyclic N_Store Algorithm builds up an envelope rate by using the n stores during the interval \( \tau_3/(n-1) \). The rate from each store is delivered for policing when the real time crosses its schedule.

The algorithm adopts a cyclic process of moving to the next store in sequence when an RM cell is received, building an envelop of rates and delivering the same at the respective schedule. All the control processes are executing independently.
The envelope building process starts whenever it receives a new explicit rate through a backward RM cell and builds up an envelope of police rates by using the rate and schedule of the store indicated by the pointer process and the new explicit rate. If the store is not empty, then it waits until the store gets emptied by the delivery process and then stores both new police rate and its schedule.

The envelope pointer process operates on three stores called 'current', 'next' and 'previous'. The 'current' and 'next' stores operate similar to 2_Store. The pointer is incremented each time it receives a new feedback from RM cell after waiting for a period of $\tau_3/(n-1)$.

The delivery process delivers the police rate from a store pointed by the pointer only when the real time crosses the corresponding police rate schedule. After the delivery, the pointer increments and points towards next store (This delivery pointer is different from envelop pointer). If the schedule is empty then that corresponding rate is not delivered and the pointer is not incremented.

The Cyclic N_Store algorithm is explained further through flow chart and C code (from the original article [2]) is reproduced in Appendix 2. The Cyclic N_Store algorithm is complicated and hence an attempt has been made to make it easy to understand through graphical representation (fig 3.3.4), flow chart (fig 3.3.4a) and the police rate tables (table 3.3.4a to 3.3.4e).
An Overview of the ABR policer

<table>
<thead>
<tr>
<th>N</th>
<th>ER</th>
<th>DEL-PR</th>
<th>NEW-PR</th>
<th>PR-SCHED</th>
<th>PACRM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ER0</td>
<td>PR0</td>
<td>PR0</td>
<td>0</td>
<td>PR0</td>
</tr>
<tr>
<td>2</td>
<td>ER1</td>
<td>PR0</td>
<td>PR0</td>
<td>0 / tER1 + t3 \rightarrow 0</td>
<td>PR0</td>
</tr>
<tr>
<td>3</td>
<td>ER2</td>
<td>PR0</td>
<td>PR1</td>
<td>0 / tER2 + t2 \rightarrow 0</td>
<td>PR1</td>
</tr>
<tr>
<td>4</td>
<td>ER3</td>
<td>PR0</td>
<td>PR2</td>
<td>0 / tER3 + t3 \rightarrow 0</td>
<td>PR2</td>
</tr>
<tr>
<td>5</td>
<td>ER4</td>
<td>PR0</td>
<td>PR3</td>
<td>0 / tER4 + t2</td>
<td>PR3</td>
</tr>
<tr>
<td>6</td>
<td>ER5</td>
<td>PR1</td>
<td>PR4</td>
<td>0 / tER5 + t2</td>
<td>PR4</td>
</tr>
<tr>
<td>7</td>
<td>ER6</td>
<td>PR2</td>
<td>PR2</td>
<td>0 / tER6 + t3 \rightarrow 0</td>
<td>PR5</td>
</tr>
</tbody>
</table>

Table 3.3.4a: N-Store police rate table built by ‘Nothing scheduled’ routine of envelope process which introduces Error 1

<table>
<thead>
<tr>
<th>N</th>
<th>ER</th>
<th>DEL-PR</th>
<th>NEW-PR</th>
<th>PR-SCHED</th>
<th>PACRM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ER1</td>
<td>PR0</td>
<td>PR1</td>
<td>0</td>
<td>PR1</td>
</tr>
<tr>
<td>2</td>
<td>ER2</td>
<td>PR0</td>
<td>PR2</td>
<td>0 / tER3 + t2</td>
<td>PR2</td>
</tr>
<tr>
<td>3</td>
<td>ER3</td>
<td>PR0</td>
<td>PR3</td>
<td>0 / tER4 + t2</td>
<td>PR3</td>
</tr>
<tr>
<td>4</td>
<td>ER4</td>
<td>PR0</td>
<td>PR4</td>
<td>0 / tER5 + t2</td>
<td>PR4</td>
</tr>
<tr>
<td>5</td>
<td>ER5</td>
<td>PR1</td>
<td>PR2</td>
<td>0 / tER6 + t3 \rightarrow 0</td>
<td>PR5</td>
</tr>
</tbody>
</table>

Table 3.3.4b: N-Store police rate table 3.3.4.a is adjusted by ‘Time sequence adjustment routine’ (TSA) and ‘Something scheduled’ routine of envelope process.

<table>
<thead>
<tr>
<th>N</th>
<th>ER</th>
<th>DEL-PR</th>
<th>NEW-PR</th>
<th>PR-SCHED</th>
<th>PACRM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ER1</td>
<td>PR3/PR6</td>
<td>PR10</td>
<td>0 / tER11 + t2</td>
<td>PR10</td>
</tr>
<tr>
<td>2</td>
<td>ER2</td>
<td>PR6</td>
<td>PR11</td>
<td>0 / tER12 + t2</td>
<td>PR11</td>
</tr>
<tr>
<td>3</td>
<td>ER3</td>
<td>PR6</td>
<td>PR12</td>
<td>0 / tER13 + t2</td>
<td>PR12</td>
</tr>
<tr>
<td>4</td>
<td>ER4</td>
<td>PR7/PR8</td>
<td>PR13</td>
<td>0 / tER14 + t3</td>
<td>PR13</td>
</tr>
<tr>
<td>5</td>
<td>ER5</td>
<td>PR6</td>
<td>PR14</td>
<td>tER4 + t2 \rightarrow 0</td>
<td>PR6</td>
</tr>
<tr>
<td>6</td>
<td>ER6</td>
<td>PR6</td>
<td>PR7</td>
<td>tER5 + t2 \rightarrow 0</td>
<td>PR7</td>
</tr>
<tr>
<td>7</td>
<td>ER7</td>
<td>PR6</td>
<td>PR8</td>
<td>0 / tER7 + t2</td>
<td>PR8</td>
</tr>
</tbody>
</table>

Table 3.3.4c: N-Store police rate table built by ‘Nothing scheduled’ routine of envelope process after the adjustment done in table 3.3.4b.

<table>
<thead>
<tr>
<th>N</th>
<th>ER</th>
<th>DEL-PR</th>
<th>NEW-PR</th>
<th>PR-SCHED</th>
<th>PACRM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ER1</td>
<td>PR6</td>
<td>PR10</td>
<td>tER11 + t2</td>
<td>PR14</td>
</tr>
<tr>
<td>2</td>
<td>ER2</td>
<td>PR6</td>
<td>PR11/14</td>
<td>tER12/14 + t2/3</td>
<td>PR14</td>
</tr>
<tr>
<td>3</td>
<td>ER3</td>
<td>PR6</td>
<td>PR12/14</td>
<td>tER13/14 + t2/3 \rightarrow 0</td>
<td>PR12/14</td>
</tr>
<tr>
<td>4</td>
<td>ER4</td>
<td>PR8</td>
<td>PR13/14</td>
<td>tER14 + t3</td>
<td>PR12</td>
</tr>
<tr>
<td>5</td>
<td>ER5</td>
<td>PR6</td>
<td>PR14</td>
<td>0</td>
<td>PR6</td>
</tr>
<tr>
<td>6</td>
<td>ER6</td>
<td>PR6</td>
<td>PR14</td>
<td>0</td>
<td>PR6</td>
</tr>
<tr>
<td>7</td>
<td>ER7</td>
<td>PR6</td>
<td>PR7</td>
<td>0 / tER7 + t2 \rightarrow 0</td>
<td>PR7</td>
</tr>
<tr>
<td>8</td>
<td>ER8</td>
<td>PR6</td>
<td>PR8</td>
<td>0 / tER8 + t2 \rightarrow 0</td>
<td>PR8</td>
</tr>
<tr>
<td>9</td>
<td>ER9</td>
<td>PR6</td>
<td>PR9</td>
<td>0 / tER9 + t2 \rightarrow 0</td>
<td>PR9</td>
</tr>
<tr>
<td>10</td>
<td>ER10</td>
<td>PR6</td>
<td>PR9</td>
<td>0 / tER10 + t2 \rightarrow 0</td>
<td>PR9</td>
</tr>
</tbody>
</table>

Table 3.3.4d: N-Store police rate table 3.3.4.c is adjusted by ‘Time sequence adjustment routine’ (TSA) and ‘Something scheduled’ routine of envelope process which introduces Error 2
An Overview of the ABR policer

### Cyclic N Store

<table>
<thead>
<tr>
<th>Police rate</th>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0</td>
<td>0</td>
</tr>
<tr>
<td>PR1</td>
<td>tER2 + ( t_3 )</td>
</tr>
<tr>
<td>PR2</td>
<td>tER3 + ( t_3 )</td>
</tr>
<tr>
<td>PR3</td>
<td>tER4 + ( t_3 )</td>
</tr>
<tr>
<td>PR6</td>
<td>tER5 + ( t_2 )</td>
</tr>
<tr>
<td>PR7</td>
<td>tER8 + ( t_2 )</td>
</tr>
<tr>
<td>PR8</td>
<td>tER9 + ( t_2 )</td>
</tr>
<tr>
<td>PR9</td>
<td>tER10 + ( t_2 )</td>
</tr>
<tr>
<td>PR10</td>
<td>tER11 + ( t_2 )</td>
</tr>
<tr>
<td>PR14</td>
<td>tER14 + ( t_3 )</td>
</tr>
<tr>
<td>PR14</td>
<td>0</td>
</tr>
<tr>
<td>PR14</td>
<td>tER14 + ( t_3 )</td>
</tr>
</tbody>
</table>

Table 3.3.4e Police rates and their schedules determined by Cyclic N Store

![Fig. 3.3.4 Police rate envelope - Cyclic N.Store algorithm](image-url)
An Overview of the ABR policer

CR: Current Rate, CT: Current rate schedule
NR: Next Rate, NT: Next rate schedule
AT: RM cell arrival time
PVR: Previous Rate, PVT: Previous rate schedule
PACR: Feedback rate ER is stored in PACR
PACRM: Maximum previous PACR
PR: Police rate, TSA: Time sequence adjustment
RSC: Rate sequence check

Fig. 3.3.4a: Simplified flow chart of Cyclic N Store algorithm

Envelop Process
Start

Something Scheduled

N PACR ≥ PACRM Y

Rate Increase

Y (AT + τi) > CT

N Scheduled New rate.

Rate Decrease

A B C D E F G

NEXT Store
NR = PACR, NT = AT + τ2

NEXT Store
NR = PACR, NT = AT + τ3

CURRENT Store
CR = PACR, CT = AT + τ2

CURRENT Store
CR = PACR, CT = AT + τ3

CURRENT Store
CR = PACR, CT = AT + τ1

CURRENT Store
CR = PACR, CT = AT + τ2

CURRENT Store
CR = PACR, CT = AT + τ3

CURRENT Store
NR = PACR, CT = AT + τ4

CURRENT Store
NR = PACR, CT = AT + τ5

CURRENT Store
NR = PACR, CT = AT + τ6

CURRENT Store
NR = PACR, CT = AT + τ7

END

N CT < PVT Y

TSA

RSC

START

N CT < PVT Y

Y CT < PVT N

END
In this example the police rate storing table stores up to 10 police rates (N=10) in sequence and then cycles around 1 to 10. For every arrival of feedback rate ER through the RM cell, a police rate PR and its schedule is stored in a new location. A store’s schedule gets cleared after the delivery of its rate to the policer. The algorithm operates on 2 stores called CURRENT and NEXT. The CURRENT store is that store which is indicated by the pointer during the envelope process. The NEXT and PREVIOUS store pointers are evaluated by the pointer process as follows:

\[
\text{NEXT} = \text{CURRENT} + 1 \quad \text{and} \quad \text{PREVIOUS} = \text{CURRENT}-1
\]

The pointer starts at CURRENT store (N=1) and then increments to N=2 at the arrival of a feedback explicit rate through RM cell. The store N=2 then becomes the CURRENT store. Obviously N=3 becomes NEXT store and N=1 is the PREVIOUS store. The evaluated new police rate (PR) and its schedule are stored either in CURRENT or NEXT store.

Refer to fig AP2.f in Appendix 2 that illustrates flow chart of the Cyclic N_Store pointer process. The pointer gets incremented at the next arrival of RM cell only if the store S = NEXT + 1 (N=4) is unscheduled (ST=0) and the CURRENT rate (CR) is not unscheduled (CT>0). If S is already scheduled (ST>0) and not yet delivered then the pointer is not incremented and the new police rate is either stored in CURRENT or NEXT store.

Refer to fig 3.3.4 and table 3.3.4a. Initially, the CURRENT store (N=1) is set to PR0 with a schedule tER0, the NEXT store N=2 is also set to PR0 but without schedule (NT=0) and the rest of the 8 stores are empty. After receiving a feedback ER1 through a backward RM cell the ‘reception process’ (Refer to fig AP2.a in Appendix 2) starts in which a ‘pointer process’ (Refer to fig AP2.a in Appendix 2.f) increments N to 2 as CT>0 and ST =0. The arrival time of the RM cell and the feedback explicit rate ER are stored in AT and PACR respectively. That means AT= real time at which RM cell has arrived carrying a feedback rate PACR=ER1.
The ‘envelope process’ starts after the reception process. This is the most complicated process in the algorithm. It is essential to understand the basic principle of the algorithm before studying a specific example. Refer to flow chart 3.3.4a and the following step by step brief description of the algorithm. This description covers only the core part of the entire algorithm as outlined in Appendix 2.

The process first checks whether the CURRENT store is unscheduled or not. If unscheduled the ‘nothing scheduled’ routine is operated otherwise the ‘something scheduled routine’ is operated. The routine operations steps are as shown below.

1. Nothing scheduled routine: The feedback rate PACR is compared with previous rate PVR and the current police rate PR to evaluate a schedule for PACR as follows.
   a. If PACR ≥ PVR and PACR > PR then PACR is scheduled at AT + τ3
   b. If PACR > PVR and PACR < PR then PACR is scheduled at AT + τ2
   c. If PACR < PVR then PACR is scheduled at AT + τ2

   PACR is stored in NEXT store (NR) and its schedule in CURRENT store (CT). PACR is stored in CURRENT store (CR) only if NT<>0.

2. Something scheduled routine: The feedback rate PACR is compared with PACR-maximum (PACRM) and the enforced police rate PR. PACRM carries the undelivered previous maximum police rate. The schedule for PACR is evaluated as follows:
   a. If PACR < PACRM then PACR is scheduled at AT+τ2
   b. If PACR ≥ PACRM but the CURRENT schedule CT is scheduled earlier than AT + τ3 then PACR is scheduled at AT + τ3
   c. If PACR ≥ PACRM and CT > AT+τ3 then PACR is scheduled at AT + τ3 provided PACR > PR otherwise it is scheduled at AT+τ2.
The police rate and its schedules in the case of 2a and 2b are stored in NEXT store (NR and NT), whereas in case 2c, it is stored in CURRENT store (CR and CT).

A common Time Sequence Adjustment (TSA) check is done after the execution of either of the routines mentioned in 1 and 2 above (Refer fig.3.3.4a). TSA is executed when the CURRENT schedule is earlier than the PREVIOUS schedule (CT<PVT).

The evaluated police rates over the arrival of the feedback rates ER1 to ER14 is given in table 3.3.4e. The envelope of the ERs and PRs are shown in fig. 3.3.4. The tables 3.3.4a to 3.3.4d show the police rates and their schedules obtained through the Cyclic N_Store algorithm after the arrival of explicit rates ER1 to ER14. Due to the complexity of the algorithm, the entire path of the algorithm (refer flow chart in Appendix 2) has to be followed step by step. However the following explanation simplifies the complexity.

Consider the feedback rates ER1 to ER6 (Refer to fig. 3.3.4 and table 3.3.4a). The police rates PR1 to PR6 and their schedules are evaluated using the ‘Nothing scheduled’ routine because initially the schedule of stores N=2 to 10 are set to zeros. Since ER1, ER2, ER3 and ER6 are increases in rates over their respective previous rates (PR0, PR0, PR1 and PR4 as per table 3.3.4a under the column NEW-PR) and the enforced police rates (PR0 and PR2 as per table 3.3.4a under the column DEL-PR), the evaluated new police rates PR1, PR2, PR3 and PR6 are scheduled after T 3 . Since ER4 and ER5 are decreases in rates over their previous rates PR2 and PR3, the new police rates PR4 and PR5 are scheduled after X 2 . PR0, PR1 and PR2 are delivered at their scheduled times ie before the arrival of ER4, ER5 and ER6. The corresponding stores N= 2, 3 and 4 are unscheduled which are marked in italics in table 3.3.4a. In table 3.34a it can be noticed that after the arrival of an ER (eg ER3) the corresponding PR (eg PR3) is stored in the next store (N=5) instead of the current store (N=4). This is an error (ERROR1) which will be examined in chapter 5).
Refer to store N=7. Since tER6+τ3 is less than tER5+τ2, the Time Sequence Adjustment (TSA) routine decrements the pointer. As such the store N=6 becomes the CURRENT store. Since the CURRENT store is already scheduled, the envelope process goes through ‘Something scheduled’ routine (Refer to fig. 3.3.4a and fig AP2.a and AP2.b in Appendix 2). Since ER6 is > PACRM (=PR5, Refer N=6 in table 3.3.4a) the algorithm passes through ‘Rate increase’ and ‘Schedule new rate first’ routines where ER6 is compared with previous rate PR3. Since ER6 < PR3 the NEXT store is unscheduled (Refer N=7 in table 3.3.4b) and process ends. TSA is not repeated because CT (tER5+τ2) > PVT (tER4+τ2).

Similarly consider the feedback rates ER7 to ER14. The police rates PR7 to PR14 and their schedules are evaluated using the ‘Nothing scheduled’ routine. Since ER7 to ER13 are decreases in rates over the enforced police rates (PR2, PR3 and PR6 (Refer to fig 3.3.4 and DEL-PR column in table 3.3.4a), the respective police rates PR7 to PR13 are scheduled after τ2. ER14 being an increase in rate over the previous rate (PR12) and the enforced police rate (PR6). The new police rate PR14 is scheduled after τ3.

Since tER14+τ3 (Refer to table 3.3.4d) is less than the schedules at N=3 (tER13+τ2) and N=2 (tER12+τ2), the ‘Time Sequence Adjustment’ (TSA) routine and ‘something scheduled’ routine pushes tER14+τ3 from store N=4 to 2 leaving the stores N= 3 unscheduled but N=4 scheduled (N=4 remains scheduled). This is an error (ERROR2) which will be examined in chapter 5).

The graphical representation of the evaluated police rates and their schedules are shown in fig. 3.3.4. It can be observed that PR1 to PR14 are wrongly scheduled due to ERROR1 and ERROR2.
An Overview of the ABR policer

<table>
<thead>
<tr>
<th>Algorithm A</th>
<th>Algorithm B 2 Store</th>
<th>Algorithm B N Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Police rate</td>
<td>Schedule</td>
<td>Police rate</td>
</tr>
<tr>
<td>PR0</td>
<td>tER0</td>
<td>PR0</td>
</tr>
<tr>
<td>PR1</td>
<td>tER1 + τ₃</td>
<td>PR1/2/3</td>
</tr>
<tr>
<td>PR2</td>
<td>tER2 + τ₃</td>
<td>PR4/5/6</td>
</tr>
<tr>
<td>PR3</td>
<td>tER3 + τ₃</td>
<td>PR10/13</td>
</tr>
<tr>
<td>PR4/5/6</td>
<td>tER4 + τ₂</td>
<td>PR14</td>
</tr>
<tr>
<td>PR7</td>
<td>TER7 + τ₃</td>
<td>6</td>
</tr>
<tr>
<td>PR8/9/10</td>
<td>TER8 + τ₂</td>
<td>7</td>
</tr>
<tr>
<td>PR11/12/13</td>
<td>TER11/12/13 + τ₂/τ₀/τ₃</td>
<td>8</td>
</tr>
<tr>
<td>PR14</td>
<td>TER14 + τ₃</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.3: Police rates and their schedules stored by Algorithm A, 2_Store and N_Store

A comparison of police rates and their schedules for Algorithms A, 2_Store and N_Store is given in table 3.3. After receiving 14 explicit feedback rates, the algorithms A, 2_Store and N_Store have evaluated police rates 8, 4 and 10 respectively. As such, Algorithm A, 2_Store and N_Store could deliver 57%, 28.5% and 71.5% of explicit rates as police rates. The 2_Store algorithm is found to be the most inefficient algorithm among the three due to its ability to deliver very few police rates.

The performance modelling of the Algorithms A, 2_Store and N_Store are described in chapter 5.
4. ABR policer performance measurement and protocol operation

4.1 Introduction

This chapter first looks at the criteria for the measurement of the performance of the ABR policer and begins by reviewing the literature in this area, then followed by a new approach to the performance measurement of the policer is introduced. The new approach involves the measurement of the percentage of tagged cells, percentage of excess cells and percentage of police rates generated by the police algorithm. It is shown in this chapter that the new approach, which is one of the major contributions of the thesis, measures the complete performance of the policer. An overview of the ABR flow control model that forms the basis for the simulation is described in chapter 5.

Section 4.2 discusses the criteria for performance measurement of the policer and reviews previous work on it. It describes a new approach in the performance measurement in the police algorithms in terms of tagged cells and excess cells. A performance analysis of all 3 algorithms (Algorithms A, 2_Store and Cyclic N_Store) is done through a graphical representation. It is shown in a typical example that the Cyclic N_Store algorithm potentially produces a significant number of tagged cells compared to Algorithm A and 2_Store which never produce tagged cells. Section 4.3 gives an overview on the ABR flow control model used for the simulation described in chapter 5.
4.2 A review of the performance measurement of an ABR policer

A network service provider's fundamental obligation for the ABR service is to support QoS commitments on compliant connection [1], particularly the cell-loss and the minimum bandwidth guarantee. Such commitments may not be possible to meet if any of the connections sharing the common link violates the traffic contract and transmits the cells at a rate higher than the feedback rate. Generally, a policer will be deployed by the service provider at the user network interface to monitor the rate of cells received by a source, discard or tag the non-conformed cells (cells received at higher than the police rate) and pass the conformed cells to the network. The policer has to detect the conformed cells accurately enough to avoid passing a large number of non-conforming cells and to wrongly tag or discard the conforming cells. Non-conforming cells from a malicious user, which if passed into the network as excess cells, will reduce the bandwidth for other users sharing the link, and may affect the guaranteed minimum bandwidth. Many such malicious users may overload the network resources and cause congestion. On the other hand, discarding or tagging conformed cells may exceed the guaranteed cell-loss to a well-behaved user. The more accurate the policer, the fewer the excess cells and tagged cells will be produced while monitoring the cells from a well-behaved source.

The ATM Forum in its TM specification [1] has specified the DGCRA for a conformance check with respect to a police rate and Algorithm A or Algorithm B (2_Store) for generating the actual police rates, which are used. The improved N_Store algorithm [2] over 2_Store has been proposed to the ATM Forum to be included in the specification. The performance assessment made on these algorithms is reviewed below.

Algorithm A is the tightest [1], as it remembers all the feedback rates. However it is complex to implement.
The 2_Store a simple algorithm but is also a flat rate policer, making it the most inaccurate policer compared to Algorithm A and N-Store algorithm policers. The 2 store (N=2) policer delivers very few rates. To an appreciable extent, the policer is almost policing a flat rate near the highest rate and remains fixed with the highest police rates. When N= 3, 5 and 12 the Number of police rates increases [11], and it does not get stuck with policing highest rates only. The improvements of the N_Store over the 2_Store policer are; 1) greater accuracy which is measured in terms of percentage of police rates generated, 2) bandwidth saving and 3) reduction of excess traffic due to malicious users.

Algorithm A is considered to be more accurate as it memorises all the feedback rates and checks the cell conformance as defined in TM specification [1]. Because of its complexity in implementation 2_Store and N-Store Algorithm B were evolved in order to simplify it. Since the simplified algorithms are still based on the conformance definition the accuracy can not be better than Algorithm A. Hence Algorithm A can be considered as perfect model (may not be an ideal) while comparing with 2_Store and N-Store algorithms in terms of accuracy.

<table>
<thead>
<tr>
<th>τ₂</th>
<th>2ms</th>
<th>10ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>τ₃</td>
<td>0.250ms</td>
<td>0.250ms</td>
</tr>
<tr>
<td>Backward RM cell inter-arrival time</td>
<td>0.09045ms</td>
<td>0.09045ms</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No. of Stores</th>
<th>Percentage of police rates delivered</th>
<th>Percentage of police rates delivered</th>
</tr>
</thead>
<tbody>
<tr>
<td>N=2</td>
<td>6.66</td>
<td>1.39</td>
</tr>
<tr>
<td>N=3</td>
<td>9.41</td>
<td>2.02</td>
</tr>
<tr>
<td>N=5</td>
<td>15.38</td>
<td>3.23</td>
</tr>
<tr>
<td>N=12</td>
<td>27.35</td>
<td>6.98</td>
</tr>
</tbody>
</table>

**Table 4.2: Percentage of rates delivered (Reproduced from [11])**

In the N_Store algorithm, if N is made sufficiently large enough to be able to memorise all the feedback rates from backward RM cells, then it is equivalent to Algorithm A. However, it is found that [11] N greater than 10
does not improve performance significantly. The N=10 algorithm is an
approximation of Algorithm A. It is recommended to the ATM Forum that
Algorithm A be adopted as the standard for policing the ABR and an
example implementation of Algorithm A can be based upon the cyclic n-
store algorithm with N=10.

The accuracy measured in terms of the number of police rates are shown in
table 4.2, reproduced from [11]. When compared to N=2, the number of
police rates are doubled for N=5 and Quadrupled for N=12. The percentage
reduces with an increase in $\tau_2$. This is because of higher rates policing for a
longer period to a maximum of $\tau_2$. The accuracy of the N-Store algorithm is
improved because the envelope-building time is reduced through the use of
N stores. The algorithm can deliver a maximum of N reference rates per
interval $\tau_3$. It operates in a manner where the delivery of these rates is fairly
evenly spaced.

All the above performance assessments have been made based on a number
of police rates that the algorithm is capable of producing. It is shown that
[11] the N_Store policer is more accurate and performs much better than the
2_Store policer. The performance measurement is based on the number of
police rates generated and the effective bandwidth saved over a period. The
measurement has not addressed the performance factors of the policer that
affect the QoS commitments made by the network. Basically, the whole
purpose of deploying a policer at a network interface is to support the QoS
commitments. Hence the performance (accuracy) measurement of a policer
can be considered to be complete only when all the three factors, such as
1.tagged cells, 2.excess cells and 3.number of police rates are measured. This
new approach is proposed in this thesis for the performance measurement of
a policer.

The algorithms used for policing should aim to achieve the best performance.
Even an ideal policer, may not be able to avoid excess cells and tagged cells
completely, due to inherent errors of Type 1 and Type 2 (Refer section 2.7).
However, a certain level of excess cells and cell-loss may be acceptable depending on the specific network. Unless this is quantified however, it may not be possible to reserve additional resources to absorb excess cells, or to guarantee a certain percentage of cell-loss to the user. Any algorithm, before deploying it as a policer, needs to be checked for its performance in terms of excess cells and cell-loss so that the network can make a feasible QoS commitments to the user. Hence it is shown here that the level of excess cells and cells loss are the main criteria for judging the performance of a policer.

With this new approach, a theoretical performance analysis of Algorithm A, B and cyclic N_Store are reviewed below. A graphical representation (for illustration purpose only) of the operation of Algorithm A, 2_Store algorithm B and Cyclic N_Store are shown in fig. 4.2.1. Refer to section 3.3 for a detailed explanation of the algorithms. The main purpose of this discussion is to understand and appreciate the importance of measuring the cell-loss and excess cells produced by each algorithm. Actual simulation results are discussed in section 4.4

4.2.1 A new method of Performance measurement through Excess and Tagged Cells

The excess and tagged cells produced by a policer directly reflects the performance of the policer responsible for causing Type 1 and Type 2 errors respectively described in section 2.6. Wrong tagging of cells sent by a well-behaved user signifies the errors in the implementation of the conformance definition [1] in the policing algorithm. These are serious errors as they incur cell loss to the user. The police rate algorithm does not tag the cells received from well-behaved sources if it is implemented as required by the conformance definition (section 3.2.2). The excess cells signify the allowance given to the user by the policer to send cells at a higher than
feedback explicit rate. The conformance definition is loosely defined to allow excess cells in order to restrict cell loss to the user under varying RTD conditions. Hence a small percentage of excess cells are absorbed in the network. However a higher percentage of excess cells may lead to network congestion. The excess and tagged cells produced by each algorithm (Algorithm A, B and N_Store) are described below with a typical example.

The fig. 4.2.1a, 4.2.1b, 4.2.1c and 4.2.1d show the envelope of explicit feedback rates along with the envelope of police rates generated by Algorithm A, 2_Store algorithm B, Cyclic N_Store and Cyclic N_Store with correction (correction 1 and 2) respectively. Fig 4.2.1d is used here mainly to explain the gaps A and B in subsequent paragraphs. The same figure is referred in chapter 5 while explaining the errors and corrections. The envelopes are generated based on the operations of algorithms described in chapter 3.

The lines joining the explicit rates ER0 to ER14 form explicit rate envelope (Refer to fig. 4.2.1a to 4.2.1d) and the lines joining the police rates from PR0 to PR14 form police rate envelope. The envelope conveys the duration for which each rate is applicable or in force. The fig. 4.2.1a to 4.2.1d show two ER envelopes, the solid line envelope is shifted by \( \tau_3 \) and dotted line envelope is shifted by \( \tau_2 \) from the original position of the envelope shown in fig. 3.3.4a to 3.3.4c in chapter 3. This shows how the source rate (after adapting ER) appears after a delay \( \tau_3 \) and \( \tau_2 \).
**Fig. 4.2.1a**

### Police rate Envelope Vs ER arrival (at Policer) rate envelop

**Algorithm A**

![Graph](image)

**Fig. 4.2.1b**

### Police rate Envelope Vs ER arrival (at Policer) rate envelop

**Algorithm B (2_Store)**

![Graph](image)
Police rate Envelope Vs ER arrival (at Policer) rate envelop
Cyclic N_Store

\( (\tau_1 = 5\text{ms and } \tau_2 = 1\text{ms}) \)

Fig. 4.2.1c

Police rate Envelope Vs ER arrival (at Policer) rate envelop
Cyclic N_Store with CORRECTION 1 and 2

\( (\tau_1 = 5\text{ms and } \tau_2 = 1\text{ms}) \)

Fig. 4.2.1d
In fig. 4.2.1d it can be observed that the policer enforces the rates PR3, PR4, PR5, PR6 and PR7 if ER7 appears at the policer after \( t_{ER7+\tau_3} \), \( t_{ER4+\tau_2} \), \( t_{ER5+\tau_2} \), \( t_{ER6+\tau_2} \) and \( t_{ER7+\tau_2} \) respectively. Thus the cells transmitted by the source at a rate ER7 may reach the policer anywhere between \( t_{ER7+\tau_3} \) and \( t_{ER7+\tau_2} \) depending upon the value of RTD at that time. If cells arrive at the policer anywhere between \( t_{ER7+\tau_3} \) and \( t_{ER4+\tau_2} \) then they are policed at a much higher rate of PR3 creating a gap A as shown in fig 4.2.1d. There is a provision for the source to increase its rate up to PR3 within the gap A. Thus the source can send excess cells to a maximum of the area of gap A. In this example the excess cells will be \( (PR3-ER7) \times [(t_{ER4+\tau_2}) - (t_{ER7+\tau_2})] \).

Similarly the cells transmitted by the source at a rate ER6 may reach the policer anywhere between \( t_{ER6+\tau_3} \) and \( t_{ER6+\tau_2} \) depending upon the value of RTD at that time. If cells arrive at the policer anywhere between \( t_{ER4+\tau_2} \) and \( t_{ER6+\tau_2} \) then they are policed at the much lower rates of PR4 and PR5 creating a gap B as shown in fig 4.2.1d. Thus the cells transmitted by the source at feedback rate ER6 are tagged by the policer as non-compliant cells. In this example the maximum number of non-compliant cells will be equal to \( PR4 \times [(t_{ER4+\tau_2}) - (t_{ER5+\tau_2})] + PR5 \times [(t_{ER5+\tau_2}) - (t_{ER6+\tau_2})] \).

The above examples are shown only with respect to one rate in each case. That is, ER7 against PR3 to show excess cells and ER6 against PR4 and PR5 to show tagged cells. Apart from this there are other rates such as ER4 to ER12 creating several gaps of A against PR3 to PR7 and ER8 to ER12 creating several gaps of B against PR8 to PR9. In fig 4.2.1a to 4.2.1d it is noticed that Algorithm A and 2_Store have only gap A whereas Cyclic N_Store has both gaps A and B.

The 2_Store algorithm has created the largest portion of gap A compared to Algorithm A and Cyclic N_Store. This is because the 2_Store has derived only 4 police rates, compared to 8 by algorithm A, 10 by Cyclic N_Store and 11 by Cyclic N_Store with correction 1 and 2. If \( \tau_2 \) had been large, then
2_Store would have generated even fewer police rates (1or 2), making it a flat rate policer. The N_Store algorithm has produced 2 police rates more than Algorithm A. However, since Algorithm A memorises all the rates, it should produce a maximum number of police rates. The 2 additional rates are due to some inherent errors in the N_Store algorithm, which will be discussed in detail in subsequent paragraphs.

In fig.4.2.1a and 4.2.1b the police rate envelope of Algorithm A and B (2_Store) never falls below the maximum of the explicit rates within a period \( \tau_2-\tau_3 \). However, for the N_Store algorithm, the region B shown in the figure falls below the Explicit rates. Region B is created because the N_Store algorithm evaluated the police rates, which are less than the maximum of the explicit rates that the algorithm received within \( (\tau_2-\tau_3) \).

The probability of the cells to arriving at the policer that fall within gap A or B due to RTD variation is low. However, the probability increases if the number of rate changes increase. Unlike gap A, gap B does not increase with an increase in \( \tau_2 \) as it is limited to the time gap between successive rates. The gap B increases with the length of the period of the rates.

The basic principle with which the ABR policer operates is dependent on the conformance definition (Refer to section 3.2) in which any increases in rates are policed after \( \tau_3 \) and decreases in rates after \( \tau_2 \). The \( \tau_3 \) and \( \tau_2 \) are minimum and maximum round trip delays. According to the conformance definition a new feedback explicit rate eg ER1 is greater than the police rate if it arrives at UNI at the time tER1 is scheduled at tER1+\( \tau_3 \) as the police rate PR1. If the subsequent feedback rate ER2 is less than ER1 then it is scheduled at tER2+\( \tau_2 \). Within the period of \( [tER2+\tau_2 - (tER1+\tau_2)] \), if all the subsequent n number of ERs are less than ER1 then all of them are scheduled after \( \tau_2 \). Thus PR1 is enforced to a period of \( (\tau_2-\tau_3) \) within which the source rates will be less than the police rate. This creates a gap between the police rate envelope and source rate envelope allowing malicious users to
utilise this gap by sending cells at higher rates to a maximum of the police rate.

The cell conformance is defined to be more lenient in order to ensure that the well-behaved user is not punished by the policer due to unpredictable RTD variations. The policer when receiving a cell from a source expects the cell to reflect the feedback rate ER sent across the UNI at time RTD ago. However if RTD varies between $\tau_3$ and $\tau_2$ after the ER crosses the UNI, then the cell cannot reflect ER at the expected time at the UNI/policing point and if the police rate is lower than the incoming cell rate then the cell becomes non-conformed. Thus the unpredictable RTD variation leads to an inherent error by creating a gap $(\tau_2-\tau_3)$ of uncertainty in choosing a police rate that exactly matches with the incoming cell rate of a well-behaved source. Hence in order to avoid this unwarranted non-conformity and cell loss for well-behaved user, the cell conformance is defined to select a police rate which is always biased towards the higher rate. The police rate algorithm has to choose the highest of all ERs received from backward RM cells arriving at the UNI/policing point within the interval $\tau_2-\tau_3$. This makes the policer loss less and avoids tagging the cells from the well-behaved source. However all the police rate algorithms (Algorithm A, 2_Store and N_Store) will create a gap between police rate envelope and source rate envelope allowing possibilities of sending excess cells into the network. This is unavoidable.

The Algorithm A memorises all the feedback rates received from RM cells and their times of arrival within the period $\tau_2-\tau_3$. It evaluates the police rate and its schedule exactly as per the conformance definition. Hence the envelope of police rate is considered as a reference for the conformance definition. The Algorithm A will neither allow more excess cells (other than that allowed due to conformance definition) nor tag the cells received from a well-behaved source. In fig 4.2.1a, the Algorithm A police rate envelope is considered as a bench mark to compare 2_Store and N_Store algorithms.
The 2_Store algorithm has created the largest gap A (fig.4.2.1b) but not the gap B. The large gap of A is due to very few police rates being generated by 2_Store algorithm which is illustrated in figs 4.2.1a to 4.2.1d and table 3.3. The 2_Store algorithm has derived only 4 police rates compared to 8 in algorithm A and 10 in Cyclic N_Store. For large value of \( \tau_2 \), the 2_Store would have produced only one or two police rates making it inefficient as a policer. The 2_Store algorithm disassociation the rate in a backward RM cell from its time of arrival at the policer. It selects a highest rate and the earliest time the rate may apply [11]. The source may not be able to adapt the rate at this earliest time. Besides it has several other inefficiencies [10] which further degrades the performance. Policing at the highest rate means allowing potentially many excess cells. However this behaviour avoids wrong tagging of compliant cells.

The N_Store algorithm, as the name suggests will have \( N>2 \) stores that allows it to memorise more than 2 feedback rates and hence delivers more police rates when compared to 2_Store. Refer to figs 4.2.1a to 4.2.1d and table 3.3. It has produced 10 police rates, 6 more than 2_Store and 2 more than Algorithm A. In spite of a greater number of police rates, it has still created gaps A and B. Gap A is less than gap B. That means it produces less excess cells compared 2_Store but wrongly tags many compliant cells. This suggests that the N_Store algorithm is little better than 2_Store, other than in producing a greater number of police rates. Hence, the conclusion on the performance of N_Store based on the generation of a large number of police rates and bandwidth saving is not sufficient to declare it as more accurate [3]. However it would appear that the, N-Store is superior to the 2_Store. Because the results do not show a performance improvement, the indication is that there are some inherent errors. The performance measurement carried out in [3] has not revealed these errors in the algorithm. Hence a new method which can evaluate an algorithm for its complete performance is needed. Such a new method is proposed in this thesis.
The performance of the ABR policer can be best judged by the amount of excess cells and tagged cells that it produces. The maximum number of excess cells that can be sent into the network without being detected by the policer is calculated as the sum of the area of gaps created between the police rate (PR) envelope and source rate (ER) envelope. The tagged cells and the excess cells can be quantified through simulation, which will be discussed in chapter 5. The quantified value will reveal how good or bad a particular algorithm is. However, Algorithm A being the perfect model, the quantity of tagged and excess cells obtained for Algorithm A may be used as reference to check the performance of 2_Store and N_Store algorithms. The quantification of tagged and excess cells not only helps to choose the best algorithm but also gives a clear picture to the network service provider to organise the network resources. The simulation is based on the ABR policer protocol operation, which is described in the next section.

4.3 The ABR policer protocol operation

In this section the ABR flow control model and the policer protocol operation, which are required to study the performance of the ABR policer is presented.

4.3.1 The ABR flow control

In the ABR service, the source traffic is controlled according to the changing network characteristics, through the feedback, which drives a flow control mechanism. In a bi-directional connection, each end system acts as source and destination. The ABR flow control occurs between a source and a destination. The feedback control information is carried by special cells called Resource Management (RM) cells. The forward RM cells flow from the source towards the destination and backward RM cells flow from their destination towards the source forming a control loop. The forward RM cells are turned around by the destination to become backward RM cells. The forward RM cells carry the current source traffic information along with the
required peak cell rate (PCR). Along the path the network may insert the
feedback control information into the RM cell directly by specifying the
explicit rate that it can support or indirectly indicate the congestion status by
setting the EFCI bit in the data cell header. The destination will also update
the feedback control information on the forward RM cell based on the EFCI
bit and the rate it can support. The destination will turn around the forward
RM cell and sends it back to its source as backward RM cell. During
congestion, the network and destination may generate backward RM cells,
insert feedback information and send it to source without waiting to turn
around the forward RM cell.

The ABR flow control mechanism is applicable to two connection types,
which are shown in fig 4.3.1.1 and 4.3.1.2. These connections may be a part
of point to point or point to multi point connections. The rate control is
applicable on all CLP equal to 0 of Data, OAM & RM cells.

Fig 4.3.1.1 End to end communication between actual source and destination

Fig. 4.3.1.2 End to end communication between virtual source and destination

SES - Sending end system, RES - Receiving end system
SN - Switch node, VS - Virtual source, VD - Virtual destination.
4.3.2 Data cell and RM cell type, structure & its fields

The types of ABR cells that come under the flow control are DATA, RM & OAM cells. All the ABR cells should be sent with CLP=0 except RM cells which are sent with CLP=1. In-rate cells are sent by the source at current cell rate (CCR). Non-conformed cells are tagged with CLP=1.

The detailed structure, fields and bit positions of a RM cell reproduced from TM specification [1] are given in Appendix 1.

A RM cell generated by the source and sent in the direction of the flow of data is called a forward RM cell. The field setting to identify a forward RM cell is DIR equal to 0 and BN equal to 0. A RM cell sent in the direction opposite to the flow of data from source is called a backward RM cell. The field setting to identify the backward RM cell is DIR equal to 1. A forward RM cell that is generated by the source, turned around, and sent back by the destination, towards the source is called a Source Generated (SG) backward RM cell. The fields setting to identify SG backward RM cell are DIR equal to 1 & BN equal to 0. If a RM cell is generated and sent towards the source by the switch or destination is called Non source generated (NSG) backward RM cell. The field settings to identify a NSG backward RM cell are DIR equal to 1 and BN equal to 1.

The information carried by a RM cell is determined using various ABR service parameters described in the next section.

4.3.3 The ABR service parameters and its negotiation.

The various ABR service parameters used in the implementation of the ABR flow control on a connection are given in the table 2.3 which is a copy of the original published in TM specification [1].
### Table 4.3: The ABR service parameters (Reproduced from TM specification [1])

<table>
<thead>
<tr>
<th>Label</th>
<th>Description</th>
<th>Units and Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCR</td>
<td>The Peak Cell Rate, PCR is the cell rate which the source may never exceed.</td>
<td>In Cells /Sec</td>
</tr>
<tr>
<td>MCR</td>
<td>The Minimum Cell Rate, MCR, is the rate at which the source is always allowed to send.</td>
<td>In Cells /Sec</td>
</tr>
<tr>
<td>ICR</td>
<td>The Initial Cell Rate, ICR, is the rate at which a source should send initially and after an idle period</td>
<td>In Cells /Sec</td>
</tr>
<tr>
<td>RIF</td>
<td>Rate increase factor, RIF, controls the amount by which the cell transmission rate may increase upon receipt of an RM cell.</td>
<td>RIF is a power of two ranging from 1/32768 to 1</td>
</tr>
<tr>
<td>Nrm</td>
<td>Nrm is the maximum number of cells a source may send for each forward RM cell</td>
<td>Power of 2 Range 2 to 256</td>
</tr>
<tr>
<td>Mrm</td>
<td>Mrm controls allocation of bandwidth between forward RM cells, Backward RM cells and data cells.</td>
<td>Constant and fixed at 2</td>
</tr>
<tr>
<td>RDF</td>
<td>The Rate Decrease Factor, RDF, controls the decrease in the cell transmission rate.</td>
<td>RDF is a power of 2 from 1/32,768 to 1</td>
</tr>
<tr>
<td>ACR</td>
<td>The Allowed Cell Rate, ACR, is the current rate at which a source is allowed to send.</td>
<td>In Cells /Sec</td>
</tr>
<tr>
<td>CRM</td>
<td>Missing RM cell count. CRM limits the number of forward RM cells which may be sent in the absence of received backward RM cells.</td>
<td>It is an integer and implementation specific.</td>
</tr>
<tr>
<td>ADTF</td>
<td>ACR Decrease time factor is the time permitted between sending RM cells before the rate is decreased to ICR.</td>
<td>Units: Seconds Range: 0.01 to 10.23 secs.</td>
</tr>
<tr>
<td>Trm</td>
<td>Trm provides an upper bound on the time between forward RM cells for an active source.</td>
<td>100 times a power of two Range: .7813 to 100 milliseconds</td>
</tr>
<tr>
<td>FRTT</td>
<td>The Fixed Round Trip Time, is the sum of the fixed and propagation delays from the source to destination and back.</td>
<td>Units - 1 microseconds Range: 0 to 16.7 seconds</td>
</tr>
<tr>
<td>TBE</td>
<td>Transient Buffer Exposure, is the negotiated number of cells that the network would like to limit the source to sending during start up periods before the first RM cell returns.</td>
<td>Units: Cells Range: 0 to 16777215</td>
</tr>
<tr>
<td>CDF</td>
<td>The Cut Off Decrease factor controls the decrease in ACR associated with CRM.</td>
<td>Zero or power of 2 in the range 1/64 to 1.</td>
</tr>
<tr>
<td>TCR</td>
<td>The Tagged Cell rate, TCR, limits the rate at which a source may send out of rate forward RM cells.</td>
<td>Constant equal to 10 cells/second</td>
</tr>
</tbody>
</table>

ATM signalling negotiates the traffic contract and QOS parameters via the call establishment messages between the end systems and the network (switches). The negotiation is done independently on both directions for bidirectional ATM connection. The main purpose is to provide a value for the parameter by the network that can support both the end systems. Since the
call establishment takes place in one round trip (source to destination and back), the negotiation must be attached to this one trip. All information exchanges needed in negotiation must take place within this one trip. The calling system proposes both a most desired and least desired value for a parameter. The negotiation starts from the most desirable, towards least desirable value. The network tries to meet the most desired value or in between or at least the least desirable one. If the network cannot meet even the least desirable value then it drops the call. If values are not specified, then the network assigns its own value.

Once the parameter values are agreed upon between the source and the network, then the called-end system (destination) receives an initial call establishment message, which has the agreed parameter values. The destination checks these agreed values with its desirable values. If the values are not met, then it may reject the call or may change its requirement as agreed by both source and network. Re-negotiation for the new parameter values can be initiated by the source even after the establishment of the call. The following parameters are negotiated and assigned a value during connection set up.

ICR, PCR, MCR, RIF, RDF, Nrm, Mrm, CRM, CDF, Trm, TCR and ADTF.

Having outlined the RM cell structure and the ABR service parameters, further discussions on the source, destination and switch behaviour appears in the next section.

4.3.4 The Source, Destination and Switch behaviours

Traffic and congestion control in the ABR service works on a set of behaviour defined for source, destination and switch (network). The behaviour for both the destination and switch provide feedback control information to the source, so it may adapt its rate to the changing needs of
the network. The source control its rate based on the feedback information. The network may deploy a policer to enforce the agreed behaviour of the source. The set of behaviour for source, destination and switch are defined in ATM Forum TM specification [1].

The source should adopt a rate not greater than ACR sent as an explicit rate feedback from the network. It can adopt a rate to a maximum of MCR if the ACR is below MCR. In case of binary congestion feedback it is specified that the source should reduce ACR to a value ACR*RDF if CI=1 and increase ACR to a value ACR*RIF if CI=0, where RDF is rate increase factor and RIF is rate increase factor.

The destination turns around the forward RM cell received from the source and returns it as a backward RM cell. Before sending the backward RM cell, the destination should update its field with congestion information (setting CI and NI bit based on EFCI bit status), the explicit rate that it can support and the direction bit to indicate it as backward RM cell. Refer to [1] for more details on destination behaviour.

The switch indicates the congestion status at the queuing point through one of the control methods, such as setting the EFCI bit in the data cell header, or setting the CI or NI bit in RM cell and updating the explicit rate field in RM cell. The switch, if generating a backward RM cell, updates all the relevant fields such as CI, NI, ER and BN. Then the cell is returned to the source. Refer to [1] for more details on switch behaviour.

4.3.5 The ABR flow control process

The main purpose is to control the sending cell rate of the source, based on the feedback received from the backward RM cell. The flow control process starts once the call is established. It is essential to understand the various
types of cells before outlining the flow control process. These are described in the next sub-section.

The source end system sends the first in-rate cell as a forward RM cell. The next in-rate cells sent are in the order of a backward RM cell (to turn around) and data cells.

4.3.5.1 End system - Sending forward RM cells, backward RM cells and forward data cells.

After the call setup, the source sends the first cell as a forward RM cell by setting fields as shown in table 4.3.8.1 below. The source should never send in rate cells > ACR. Also, the ACR should never exceed PCR.

Initially the ACR is set to ICR. The source should set CCR to whatever rate it sends the cells. CCR should satisfy the equation MCR \leq CCR \leq ICR. If CCR is less than the Tagged cell rate(TCR), then RM cells may be sent at a rate \leq TCR.

<table>
<thead>
<tr>
<th>ATM Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>RM - VPC; VCI = 6 and PTI = 110</td>
</tr>
<tr>
<td>RM - VFC: PTI = 110</td>
</tr>
<tr>
<td>CLP = 0/1</td>
</tr>
<tr>
<td>RM Protocol Identifier</td>
</tr>
<tr>
<td>ID = 01</td>
</tr>
<tr>
<td>Message Type</td>
</tr>
<tr>
<td>DIR=0, BN=0, CI=0, NI=0, RA=0</td>
</tr>
<tr>
<td>ER = PCR</td>
</tr>
<tr>
<td>CCR = The rate at which the cell is sent</td>
</tr>
<tr>
<td>MCR = MCR</td>
</tr>
<tr>
<td>QL = 0</td>
</tr>
<tr>
<td>SN = 0</td>
</tr>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>Reserved (6 bits) + CRC (10 bits)</td>
</tr>
</tbody>
</table>

Table 4.3.5.1: The RM cell field settings
The Sending of subsequent forward RM cells is subjected to the following. If TELR is the Time Elapsed since the Last in-rate forward RM cell, then next in rate RM cell should be sent only if

Number of in-rate cells sent within TELR is at least $M_{rm}$ and $TELR > T_{rm}$

OR

Number of in-rate cells sent within TELR is at least $N_{rm} - 1$

Where $M_{rm}$ and $N_{rm}$ are respectively the Minimum and Maximum number of cells between RM cell generation and $T_{rm}$ is the upper bound on the time between successive forward RM cells.

If a new rate is adopted then the first cell with the new rate should be a RM cell and then subsequent in rate cells follow the new rate.

Before sending a forward RM cell, the adaptation of a new rate under various conditions should be followed as given below.

i) If $TELR > A_{DTF}$ and $ACR > ICR$

Then $ACR$ should be equal to $ICR$. $A_{DTF}$ is the $ACR$ decrease time factor.

ii) The $ACR$ obtained in i) should be further adjusted as per the conditions given below.

If the number of forward in-rate RM cells sent since the arrival of last backward (source generated) RM cell is $\geq CRM$ then $ACR$ is further reduced to $ACR = ACR - ACR*CDF$. Where $CDF$ is Cut off decrease factor and $CRM$ is Missing RM cell count.

If the reduced $ACR$ results in $< M_{CR}$, then the new rate will be $ACR = \text{Max} (ACR, M_{CR})$
iii) When a backward RM cell is received, then the new rate ACR is adjusted as per the conditions below and the first forward RM cell is sent with the new rate then subsequent in rate cells should follow the new rate.

iv) If CI=0 & NI=0 and PNI = 1 and if condition i) above is not true after the arrival of the previous backward RM cell then,

• increase ACR to \( ACR = ACR + RIF \times PCR \)
  
  If ACR exceeds PCR then \( ACR = \min(ACR, PCR) \)
  
  If ACR exceeds ER then \( ACR = \min(ACR, ER) \)
  
  If ACR reduces below MCR then \( ACR = \max(ACR, MCR) \)

• Otherwise, do not increase ACR.

v) If CI = 1 then,

• decrease ACR to \( ACR = ACR - ACR \times RDF \)
  
  If ACR exceeds ER then \( ACR = \min(ACR, ER) \)
  
  If ACR reduces below MCR then \( ACR = \max(ACR, MCR) \)

The new rate is calculated at source based on the feedback information obtained from backward RM cell before sending a forward RM cell. The backward RM cell (self generated) will be sent only when no forward RM cell and or data cells are waiting to be sent. Preference over waiting RM and data cells will be given to turn around a backward RM cell received from the far end system. Data cells are sent when no forward and backward RM cells are waiting to be sent.
4.3.5.2 End system - Receiving forward RM cells, backward RM cells and forward Data cells

The destination end system receives the forward RM cells sent by the source, then returns them as illustrated above. The backward RM cell received by the source is either its own forward RM cell turned around by the destination, or an RM cell generated by switch or destination.

The forward RM cell sent by the source passes through switches in both forward and backward directions. Switches supporting the RM cell may set CI to 1 if the node is congested, NI to 1 for impending congestion and may reduce the ER to a value that it can support. When the destination receives a forward RM cell sent by source then it sets the CI bit field within the RM cell to 1 only if it had received EFCI bits as 1 from data cells received prior to RM cell. It may also reduce the ER to a value that it can support. After setting the field with direction bit DIR to 1, the destination turns around the cell and transmits either in rate with CLP=0 or out of rate with CLP=1. No switch or destination should change the CI bit to zero and increase an ER value.

If a second forward RM cell is received from an end system (destination) to turn around while the first turn around RM cell is already scheduled for transmission, then the destination has two choices in sending the turns around RM cells.

- The first choice will be to send the first turn around RM cell, which is due for transmission after replacing the contents from the second one.

- The second choice will be to send the second turn around RM cell after setting the fields and discard the first one.

If the source receives a Backward RM cell then it takes the feedback information from the following fields DIR, CI, NI, BN and ER, then reflects
this information in the next forward RM cell, as mentioned in the source behaviour given in section 4.3.4.

When the data cells are received by the destination, it saves the EFCI status bit taken from them. When the data cells pass through switches, any congested switch node will set the EFCI bit to 1.

The ABR flow control discussed in this section, is the basis of the simulation model described in the following chapter.
5. ABR policer Simulation

5.1 Introduction

This chapter begins with an overview of the ABR flow control model that forms the basis for the simulation described in section 4.3.5. This is followed by the performance results of policing mechanisms that involve 3 different police rate algorithms A, 2_Store and N_Store are analysed and discussed. In particular, the inefficiencies and errors in the cyclic N_Store algorithm are identified. The corrections that overcome the inefficiencies are discussed in chapter 6. Improvements over the existing N_Store algorithm and a new simplified N_Store_I algorithm that can overcome the inefficiencies are also described in chapter 6.

Section 5.2 presents simulation models of the ABR source, destination, ERICA switch model and the ABR police model (using DGCRA and Algorithms A or 2_Store or Cyclic N_Store). Section 5.3 describes and lists out parameter values and set of assumptions made in the simulation and provides rationale for each of them. Section 5.4 analyses the performance results of all the three algorithms (Algorithms A, 2_Store and Cyclic N_Store) and identifies the inefficiencies in the algorithms, particularly the N_Store algorithm. Section 5.6 concludes with the outcome of the results and discusses the improvements needed in the N_Store algorithm.
The simulated ABR flow control Model is shown in fig. 5.1.1. This is based on the ABR rate based flow control mechanism described in the previous section 4.3.

5.1.1 User Network Interface (UNI) model

The UNI model comprises a single ABR source, an Explicit Rate Indication for Congestion Avoidance (ERICA) [12] switch and an ABR policer which are further described in detail in subsequent sections. A brief description of the operation of UNI model is given below.

Fig.5.1.1 explains the model graphically. The ABR source initially starts transmitting the forward cells at the ICR (Initial Cell rate) until it receives feedback from the switch via the backward RM cells. The backward RM cells transmitted across the UNI take a time delay of \( t_{d1} \) to reach the source. The Source sends the first cell immediately after adopting the new rate based on the feedback explicit rate ER received from the backward RM cell. The first cell with the new rate takes a time delay of \( t_{d2} \) to reach the UNI/policer. Generally it is assumed that \( t_{d1} = t_{d2} \) is the sum of one way propagation delay, transmission delay and queuing delay (if there are intermediate non-ATM switches and routers) between user and the policer at UNI. The RTD= \( t_{d1} + t_{d2} \) is assumed to vary between \( \tau_2 \) and \( \tau_3 \).
The source transmits a RM cell after every $N_{rm}$ number of forward cells. After receiving each of the feedback rates $ER_1$ to $ER_4$, the source adopts and transmits the cells at corresponding rates $CCR_1$ to $CCR_4$. The policer after receiving each cell checks the cell compliance with respect to new rate (new reference inter-cell interval $I(k)$) using the DGCRA algorithm and one of the police rate algorithms (Algorithm A or 2_Store or cyclic N_Store) as described in chapter 3. Here all the three algorithms are considered in order to compare their performance. The policer expects the arrival of forward cells with the new rate after a round trip time delay that is anywhere between $\tau_3$ and $\tau_2$ and starts enforcing the new rate as per the conformance definition (refer section 3.2.2). The policer tags the non-conformed cells and passes
only the conformed cells into the network. It is assumed that the tagged cells are discarded and the conformed cells are passed on to the network.

The ERICA switch node evaluates the new rate ER after receiving a fixed number of N conformed cells or after a fixed timeout T. The ERICA switch records the evaluated new ER value in all backward RM cells. It is assumed that the switch itself acts as a virtual destination and hence it turns around the RM cell and sends it as a backward RM cell across UNI towards the source. Before it crosses the UNI, the ER value and the arrival time is copied by the policer.

The Source receives the backward RM cells after a delay of $t_d$. It copies the ER value to ACR, which is its maximum allowed cell rate from that time. It evaluates a current cell rate CCR to be less than or equal to ACR and sends the cell due for transmission at CCR. Generally CCR is equal to ACR. It sends the forward RM cell after storing the new CCR value in it. The policer checks the conformance of the received cells at a police rate PR that it evaluated using one of the three Algorithms A or B (2_Store) or N_Store.

Followed by the explanation above of a generalised background on the model, a more detailed examination particularly on the source and its rate control, the ERICA switch that allocates a fair share of bandwidth and the policer that evaluates a new police rate and its schedule on receiving a new RM cell is given below.

### 5.1.2 Source model

The simulation uses a single ABR source and 10 VBR sources. Both share a common link of a capacity of 150Mbps. The VBR traffic, which is given the first priority, will occupy the link bandwidth allowing the remaining bandwidth to be used by the ABR source. Thus through dynamic changes in VBR traffic the available bandwidth for ABR service also changes...
The ABR policer Simulation

dynamically and hence the ABR source rate. The following assumptions are made.

- Only the ER mechanism is modelled.
- The ABR source is a persistent (greedy) source in which the cells are continuously generated.

The source begins cell transmission at the initial cell rate (ICR) until it adopts an explicit rate from a backward RM cell. The explicit rate (ER) becomes the source’s Allowed Cell Rate (ACR) which is the maximum rate at which the source can send cells from that time. The source evaluates its Current Cell Rate (CCR) and transmits subsequent cells at that rate until it receives a feedback. The source generates a forward RM cell after every Nrm-1 forward cells or Trm period, whichever occurs first.

Inputs to source model are 1. Nrm and Trm provided from input file. 2. ER value from the backward RM cell and its arrival time. 3. ICR, MCR and PCR from the input file.

Source model outputs are 1. Adopted source rate CCR and the transmission time 2. A count of cells transmitted.

10 ON/Off VBR sources generate VBR traffic. The ON and Off periods of each VBR source are independently varied in an exponentially random way. The maximum bandwidth used when all the 10 VBR sources are ON is 80% of the target rate. The VBR traffic in terms of percentage of link target is the output. It is zero during Off periods.

5.1.3 Switch model

The ERICA [13] [14] scheme is the switch node model used in the simulation. The model works as follows. The target rate is evaluated by multiplying the utilisation factor U with link rate C (target rate = U*C). U is
The ABR policer Simulation

The ABR policer Simulation generally chosen as 0.9 to keep the target rate just below link capacity. The target rate is the maximum rate allowed to be shared by all the sources so as to keep the total utilised bandwidth just below the link bandwidth. However U can be up to 100%. Using the target rate, the ERICA scheme evaluates the available bandwidth for the ABR source, based on the available bandwidth. It then calculates the received traffic at UNI from the ABR source based on either 'No' number of cells received from the source over a fixed period 'To' or fixed number of cells N over a period Tn. A load factor is evaluated by dividing the incoming rate by the target rate. Dividing the source rate CCR by the load factor will provide a new ER value for that source. The Switch calculates the arrival rate 'N/Tn' after collecting N cells over a period 'Tn'. If N cells do not arrive within a timeout period 'To' then the rate is calculated as 'No/To' where 'No' is the number of cells received within the period 'To'.

The switch calculates the following parameters to finally evaluate ER.

- ABR capacity = (target rate - VBR traffic).
- Load factor (overload or under-load) = Incoming cell rate / ABR capacity.
- Fair share = ABR capacity / number of active ABR sources.
- VC share = incoming rate / load factor.
- ER = Maximum [Minimum (PCR, VC share), MCR].

The switch inserts this ER value into all backward RM cells which cross the UNI until another ER value is evaluated.

In this thesis the simulation model has a single ABR source and hence the values, of the ABR capacity, the fair share and VC share are equal and are sent as explicit rate feedback ER to source.

Inputs to the ERICA switch are 1) Conformed cells count and their arrival times provided by the policer. The maximum counts threshold ‘N’ and time out period 'switch_meas_intvl' are provided from input file. 2) The VBR
traffic in percentage of link target. 3) Link capacity C, PCR, MCR and Utilisation factor U are provided from input file

The output from the ERICA switch is the new ER value and its generation time, which is inserted into the backward RM cell.

5.1.4 Destination model

In this simulation model the switch itself acts as virtual destination. It receives the forward cells and RM cells from the source. RM cells are returned to the source as backward RM cells.

5.1.5 The ABR policer model

The ABR policer is modelled as a combination of cell conformance check algorithm DGCRA and one of the Police rate and schedule algorithms (Algorithm A, 2_Store Algorithm B and N_Store). The DGCRA is exactly modelled as per virtual schedule algorithm specified [1] for ER conformance check (see section 3.3.1). The Algorithm A is modelled as specified in TM4.0 [1], which is described in section 3.3.2. The 2_Store algorithm B is modelled as per the pseudo code given in TM4.0 [1] (refer section 3.3.3). The N_Store algorithm is modelled as per the pseudo code [3].

- The conformance check and Police rate algorithms run independently.
- Initially the ICR is assigned as a police rate and the DGCRA algorithm checks the incoming cell rate with respect to the ICR.
- The police rate algorithm evaluates a new police rate PR and its enforcing schedule whenever it receives a backward RM cell. The algorithm uses the time that it received the backward RM cell and the value of ER that the RM cell contains.
- The police rate is delivered to conformance check when the real time crosses the police rate-enforcing schedule. As a process of delivery the PR is copied on to PACR of DGCRA algorithm.

- The DGCRA algorithm simply checks the inter-cell interval of incoming cells comparing to \( I = 1 / \text{PACR} \). PACR is totally transparent to DGCRA.

- The DGCRA after conformance check will increment a counter for the occurrence of each conforming and non-conforming status.

Inputs to the police rate algorithms are; 1) The Explicit Rates (ER) and the RM cell arrival times generated by ERICA switch. 2) The \( \tau_2 \) and \( \tau_3 \) parameters from the input file. The outputs of the police rate algorithm are the police rates and their schedules.

Inputs to DGCRA algorithm are; 1) The police rate PACR delivered by police rate algorithm. 2) The ICR, PCR, MCR and CDVT from input file. 3) Forward cell arrival time. The outputs of DGCRA algorithm are the incremented conformance and non-conformance counters.

### 5.1.6 RTD variation and choice of \( \tau_2 \) and \( \tau_3 \)

The RTD, \( \tau_2 \) and \( \tau_3 \) are critical factors in determining the performance of the ABR policer. The RTD variations are mainly influenced by the VBR and CBR traffic (see section 3.2). Hence the RTD is expected to vary throughout the connection time, depending upon these traffic variations. The minimum RTD is the fixed propagation delay. The maximum RTD is the peak value of the variations, which are influenced by the VBR and CBR traffic load (queuing delays), number of hops in the connection and number of sub-networks involved in the connection. Delays in one sub-network [9] may vary significantly due to network operators reducing the level of the
congestion threshold or increasing the amount of bandwidth available to the ABR class. It is found that the end to end delay for the ABR service increases [16] more with VBR traffic than with CBR traffic sharing the common link. The end to end delay further increases with the number of hops.

With VBR traffic ON (for a throughput of 60% link capacity) the end to end delay within the simulation is found to increases approximately by 1milli seconds, 5milli seconds and 10milli second for LAN (1hop), MAN (2hop shorter link) and WAN (2hop longer link) respectively. For LAN, MAN and WAN environment, the propagation delay is assumed to be in the order of 0.1milli seconds, 1milli second and 5milli seconds for corresponding link lengths of 25km, 250km and 1250km respectively.

The exact nature of RTD variation is difficult to predict due to the influence of several factors. So far no literature in this regard has been found. Hence in the simulation model, the following 3 types of RTD variations are assumed. These encompass a wide range of practical RTD scenarios. The RTD is varied between $\tau_2$ and $\tau_3$ values.

1. RTD varies dynamically in accordance with VBR traffic variation.

2. RTD varies randomly between $\tau_2$ and $\tau_3$. The RTD value is given by $\text{RTD} \rightarrow (\tau_2 - \tau_3)\times \text{random number}$. The random number varies between 0 and 1. RTD is initially set to $\tau_3$.

The RTD is varied randomly for such scenarios in which the unpredictable variations in the traffic and hence the delay occurs.

3. RTD varies randomly but with a ramp in the increases and decreases. The RTD value is given by

$$\text{RTD} \rightarrow \text{Ramp factor} \times (\tau_2 - \tau_3) \times \text{random number}.$$  

The ramp factor is chosen between 0.1 to 0.5 and RTD is initially set to $\tau_3$.  

The ramping of random variations in RTD is included mainly for those scenarios in which the random variations are not jumpy (consecutive steep increases and decreases) but smooth.

Ideally the policing rate changes are directly associated with the RTD. Exact RTD variation in the actual scenario is very difficult to measure or predict. As an ideal case it is possible to simulate the RTD variations and hence the source rate epochs. However analysing the results obtained from such an ideal case may not provide significant contributions towards the improvement of the algorithm. Hence an attempt has been made to adopt more practical variations of RTD (3 variations as mentioned above), \( \tau_2 \) and \( \tau_3 \).

The \( \tau_2 \) and \( \tau_3 \) are the limits on the upper and lower bounds on the variation of RTD. Choice of \( \tau_2 \) and \( \tau_3 \) as explained in section 3.2, is set by the network operator. The two-way propagation delay between the source and policing point may be taken as \( \tau_3 \) in which the delay variations over the fixed propagation delay is zero. The determination of \( \tau_2 \) is difficult because the RTD variations are unpredictable and hard to measure. The choice of \( \tau_2 \) directly influences the performance of a policer. Larger values of \( \tau_2 \) provide opportunities to malicious users, which can upset the QoS commitments made to other users by the network. Similarly the choice of the small value of \( \tau_2 \), that is, less than the actual RTD, may penalise the well-behaved users, which again upsets the QoS commitments made by the network. Hence the network operator has to strike a balance between the cell loss guarantee and the available resources on the network while choosing \( \tau_2 \).

The whole conformance definition is indeed oriented towards a more liberal policer [14] due to the unpredictable variation of RTD, which is always biased towards a higher rate, to avoid cell-loss to the well-behaved user. Network operators need to tune their buffers in order to absorb the excess cells arising out of the conformance definition. Hence it is advisable to choose larger values of \( \tau_2 \). An approximate value of \( \tau_2 \), may be obtained...
based on the statistics of maximum end to end delays [16] for a connection in the presence of VBR and CBR traffic and initial measurement of RTD during call setup. The delay can be further tuned to be more accurate in subsequent connections on the same path based on the previous experience.

In the simulation, the following values of $\tau_2$ and $\tau_3$ are assumed for three-network link scenarios.

1. For LAN environment a one-hop network reference model [16] as shown in fig. 5.1.6.1 is considered. The one-way propagation delay of 0.05 milli seconds (4 micro sec. per km) for a distance of 12.5 kms (12.5 kms for 1st Hop) is considered. The $\tau_3$ and $\tau_2$ are assumed to be 0.1 milli seconds and 1 milli second respectively.

2. For MAN environment a two-hop network reference model as shown in fig. 5.1.6.2 is considered. The one-way propagation delay of 0.5 milli seconds for a distance of 125 kms (25 kms for 1st Hop + 100 kms for 2nd Hop) is considered. The $\tau_3$ and $\tau_2$ are assumed to be 1 milli second and 5 milli seconds respectively.

3. For WAN environment a two-hop network reference model is considered as shown in fig. 5.1.6.2. The one-way propagation delay of 2.5 milli seconds for a distance of 625 kms (25 kms for 1st Hop + 600 kms for 2nd Hop) is considered. The $\tau_3$ and $\tau_2$ are assumed to be 5 milli seconds and 15 milli seconds respectively.
Fig. 5.1.6.1: 1 Hop Reference network Model [16]

Fig. 5.1.6.2: 2 Hop Reference network Model [16]
5.2 Assumptions and Parameter values

A discrete event simulation has been developed in 'C'. The simulation mirrors the ABR performance model described in section 5.1.

The various assumptions and input parameter values that are used in the simulation model are given in table 5.2.1. The performance analysis is based on the assumptions and parameter values shown in the table 5.2.1.

We see the performance of an ABR policer located at one and two-hop ATM link [figs. 5.1.6.1 and 5.1.6.2] network interfaces. The policer is placed in such an environment to police one ABR connection. The PCR, MCR and ICR, which are determined by the network for the ABR connection, are set to typical values of 150Mbps, 0.15Mbps and 1.5Mbps respectively. The ERICA switch determines the explicit rate (ER). The switch utilisation factor is set to 90% and averaging interval to 100 cells or 1milli second (which ever occurs first). The values for RTD, \( \tau_2 \) and \( \tau_3 \) are set as shown in the table 5.5.1. These are found to be the typical end to end delay variations for the ABR connection in the presence of VBR traffic [16]. Another important factor that influences the performance of the N_Store policer is the value N, the number of stores. \( N=10 \) is found to be the limit [3] in obtaining the maximum performance from the algorithm. This is illustrated in figs. 5.5.1 to 5.5.3. The results in terms of cell-loss, tagged cells and percentage police rates are obtained in WAN environment for N-Store policer for various values of N. The results vary for \( N < 10 \) but remain stabilised for \( N \geq 10 \). Hence the value of \( N_{\text{Store}}=10 \) is used in the simulation for the N_Store algorithm.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description and Assumptions</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCR</td>
<td>Peak cell rate = 150.0 Mbps</td>
<td>353.773 cells/milli second</td>
</tr>
<tr>
<td>MCR</td>
<td>Mean cell rate = PCR/1000</td>
<td>0.353773 cells/milli second</td>
</tr>
<tr>
<td>ICR</td>
<td>Initial cell rate in = PCR/10</td>
<td>35.3773 cells/milli second</td>
</tr>
<tr>
<td>ABR Source</td>
<td>Assumptions: Single greedy source. Zero transmission delay. Flow control through Explicit Rate (ER) feedback.</td>
<td></td>
</tr>
<tr>
<td>CCR</td>
<td>Current Cell Rate</td>
<td>Initially set to ICR CCR adopts feedback rates received from network via backward RM cells.</td>
</tr>
<tr>
<td>VBR Source</td>
<td>Assumptions: 10 ON/OFF sources. ON/OFF periods vary exponentially random way.</td>
<td></td>
</tr>
<tr>
<td>Vbr_traffic</td>
<td>Total VBR traffic generated from 10 sources</td>
<td>VBR traffic varies from 0 to 80% of target rate = 108Mbps</td>
</tr>
<tr>
<td>ON/OFF periods</td>
<td>ON or OFF period of each VBR source</td>
<td>Varied exponentially random way between 0 to 2 milli sec. with a mean of 1 milli sec.</td>
</tr>
<tr>
<td>ERICA switch</td>
<td>Assumptions: Takes only conformed cells for feedback rate calculation.</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>Link rate = 150.0 Mbps</td>
<td>0.353773 micro secs</td>
</tr>
<tr>
<td>U</td>
<td>target ratio or Link utilisation used in ERICA switch algorithm</td>
<td>90%</td>
</tr>
<tr>
<td>target_Rate</td>
<td>This is the maximum bandwidth shared between VBR and ABR traffics</td>
<td>target_Rate=C*U=135Mbps</td>
</tr>
<tr>
<td>ABR_traffic</td>
<td>Available ABR bandwidth</td>
<td>27Mbps to 135Mbps</td>
</tr>
<tr>
<td>Nrm</td>
<td>Limit on number of forward cells to be transmitted before an RM cell.</td>
<td>31 Cells</td>
</tr>
<tr>
<td>Trm</td>
<td>Time out in micro secs. for transmitting 1 forward RM cell if Nrm cells couldn't be transmitted within this time.</td>
<td>1 milli sec.</td>
</tr>
<tr>
<td>N</td>
<td>Fixed Number of cells collected by switch node over a period T to find average source rate as per ERICA switch algorithm</td>
<td>100 Cells</td>
</tr>
<tr>
<td>Switch_meas_in_tvl</td>
<td>Time out for receiving N cells at switch node. This is used in measuring incoming cell rate.</td>
<td>1000 micro secs.</td>
</tr>
</tbody>
</table>
### ABR Policer

<table>
<thead>
<tr>
<th>N_Store</th>
<th>Number of stores in N_Store algorithm</th>
<th>10 stores</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\tau_2)</td>
<td>Maximum RTD used by the policer for conformance check. The (\tau_2) is varied from</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LAN = 0.1 to 1milli sec. In steps of 0.1 milli sec.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAN = 1 to 5milli sec. In steps of 0.4 milli sec.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WAN = 5 to 15milli sec. In steps of 1 milli sec.</td>
<td></td>
</tr>
<tr>
<td>(\tau_3)</td>
<td>Minimum RTD used by the policer for conformance check</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LAN = 0.1 milli sec.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAN = 1 milli sec.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WAN = 5 milli sec.</td>
<td></td>
</tr>
<tr>
<td>RTD_ini</td>
<td>Initial Round Trip Delay</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LAN = 0.1 milli sec.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAN = 1 milli sec.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WAN = 5 milli sec.</td>
<td></td>
</tr>
<tr>
<td>RTD</td>
<td>Round Trip Delay variation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RTD varies between (\tau_2) and (\tau_3) values as per 3 variation factors.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1. VBR sources ON/OFF variation factor (0 to 1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Random number (0 to 1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Random number (0 to 1) Ramping factor (0.1 to 0.5)</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2.1: The typical parameter values used in the simulation of network with the ABR policer.
**FIG. 5.2.3**

Value of N (number of stores)

% of Tagged Cells generated when cells are transmitted by source at feedback rates

% of Excess Cells generated when cells are transmitted by source at Police rates

WAN Environment

Effect of N over % of Tagged Cells

N Store Algorithm

**FIG. 5.2.2**

Value of N (number of stores)

% of Excess Cells generated when cells are transmitted by source at Police rates
The ABR policer simulation
A systematic statistical procedure is used to determine confidence intervals on the results obtained. The results have been verified for a width of 95% confidence interval to be less than 5%.

5.3 Results:

The results for Algorithms A, 2_Store and N_Store were obtained through the above simulation model. The Figs. 5.3.1 to 5.3.3 show the simulation results for LAN, MAN and WAN environment with the RTD varying with VBR traffic. Each figure shows 3 graphs called NS_O_C0, NS_O_C1 and NS_O_C2, which stands for N_Store Algorithm with no (zero) correction, with correction for ERROR1 and with correction for ERROR1 and ERROR2 respectively. In all the 3 graphs one can find that the curves NS_O_C0 and NS_O_C1 are erratic and shows abnormal values. This, after thorough analysis was found to be due to two errors in the algorithm, which are described in the section below. The results of the N_Store algorithm can be compared with those of Algorithm A and 2_Store only after rectifying the errors and obtaining comparable results.

5.3.1 Result analysis and errors in N_Store algorithm

The N_Store algorithm (in the form of C code) that was published [3] in an ATM contribution paper is used in this thesis for the performance analysis. After analysing the algorithm based on the results obtained NS_O_C0 (fig 5.3.1 - 5.3.3) it is found that the algorithm has two basic errors ERROR1 and ERROR2.
Performance of Policing in terms of % Tagged Cells
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 5.3.1

Performance of Policing in terms of % Tagged Cells
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 5.3.2

Performance of Policing in terms of % Tagged Cells
In WAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 5.3.3
These errors are actually the cause for gaps A and B (fig. 4.2.1d) which are created over the envelope of police rates. The ERROR1, ERROR2 and the corrections on these errors are discussed in the subsequent sections. Refer to section 3.3.4 where the operation of Cyclic N-Store algorithm is described in detail and the errors in the algorithm are indicated. Use the tables 3.3.4a to 3.3.4e in conjunction with the text in sections 5.3.1.1 to 5.3.1.4 to fully appreciate the arguments in those sections.

5.3.1.1 ERROR1

This is the error that occurs in the algorithm while scheduling a new police rate when nothing is scheduled. The ERROR1 is introduced by the ‘Nothing Scheduled’ routine. Fig 5.3.1.1 shows part of the algorithm where an error is introduced.

**ENVELOPE PROCESS - Nothing Scheduled**

- **Nothing Scheduled**
  - NR=PACR
  - If NT <> 0
    - CR=PACR
    - PACRM=PACR
    - Delay the Rate
    - CT=AT+τ2
  - If PACR>PVR
    - Don’t Delay the Rate
    - CT=AT+τ3
  - If PACR>PR
    - CT=AT+τ3

Fig. 5.3.1.1
It can be noticed that the PACR (carries feedback rate ER received from backward RM cell) is stored in NEXT store (NR=PACR) and schedule is stored in CURRENT store (either CT = AT+τ2 or CT = AT+τ3). The PACR is stored in CURRENT store only when NT<>0.

However, the envelope process (after the store being incremented by the pointer process) enters into the ‘Nothing Scheduled’ routine only when it finds CT=0 (refer AP2.a in Appendix 2). Previous to the store increment, the pointer CT was NT and NT was ST. Since the pointer process demands ST=0 before the increment (refer AP2.f in Appendix 2), obviously ST becomes NT after the increment and hence NT=0. This argument substantiates that the ‘Nothing Scheduled’ routine will never have a condition of NT<>0. That means PACR will never be stored in CR. This creates an abnormality of storing a new police rate in NEXT store but scheduling it in the CURRENT store leading to wrong scheduling. This is ERROR1.

<table>
<thead>
<tr>
<th>Rate</th>
<th>Schedule</th>
<th>Rate</th>
<th>Schedule</th>
<th>Rate</th>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0</td>
<td>0</td>
<td>PR0</td>
<td>0</td>
<td>PR0</td>
<td>0</td>
</tr>
<tr>
<td>PR1</td>
<td>τER2+τ3</td>
<td>PR1</td>
<td>τER1+τ3</td>
<td>PR1</td>
<td>τER1+τ3</td>
</tr>
<tr>
<td>PR2</td>
<td>τER3+τ3</td>
<td>PR2</td>
<td>τER2+τ3</td>
<td>PR2</td>
<td>τER2+τ3</td>
</tr>
<tr>
<td>PR3</td>
<td>τER4+τ3</td>
<td>PR3</td>
<td>τER3+τ3</td>
<td>PR3</td>
<td>τER3+τ3</td>
</tr>
<tr>
<td>PR6</td>
<td>τER5+τ3</td>
<td>PR4</td>
<td>τER4+τ3</td>
<td>PR4</td>
<td>τER4+τ3</td>
</tr>
<tr>
<td>PR7</td>
<td>τER8+τ2</td>
<td>PR5</td>
<td>τER5+τ2</td>
<td>PR5</td>
<td>τER5+τ2</td>
</tr>
<tr>
<td>PR8</td>
<td>τER9+τ2</td>
<td>PR6</td>
<td>τER6+τ2</td>
<td>PR6</td>
<td>τER6+τ2</td>
</tr>
<tr>
<td>PR9</td>
<td>τER10+τ2</td>
<td>PR7</td>
<td>τER7+τ2</td>
<td>PR7</td>
<td>τER7+τ2</td>
</tr>
<tr>
<td>PR10</td>
<td>τER11+τ2</td>
<td>PR8</td>
<td>τER8+τ2</td>
<td>PR8</td>
<td>τER8+τ2</td>
</tr>
<tr>
<td>PR14</td>
<td>τER14+τ3</td>
<td>PR9</td>
<td>τER9+τ2</td>
<td>PR9</td>
<td>τER9+τ2</td>
</tr>
<tr>
<td>PR14</td>
<td>0</td>
<td>PR13</td>
<td>τER13+τ3</td>
<td>PR13</td>
<td>τER13+τ3</td>
</tr>
<tr>
<td>PR14</td>
<td>τER14+τ3</td>
<td>PR14</td>
<td>τER14+τ3</td>
<td>PR14</td>
<td>τER14+τ3</td>
</tr>
<tr>
<td>PR14</td>
<td>0</td>
<td>PR14</td>
<td>τER13+τ3</td>
<td>PR14</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PR13</td>
<td>τER13+τ3</td>
<td>PR13</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.3.1: Police rates and their schedules stored by N_Store before and after the correction of ERROR 1.
The police rates and their schedules as given under the column NS_O_C0 in the table 5.3.1 where PR1 to PR10 are wrongly scheduled from the arrival times of tER2 to tER11 respectively. The ‘Nothing Scheduled’ routine is generally executed under the situation of current (CT) and next (NT) stores being unscheduled (ie CT=0 and NT=0) which occur during start up or when the stores are cleared by the delivery process.

5.3.1.2 ERROR2

The second error is in the Time Sequence Adjustment (TSA) process is illustrated through the flow chart in figures AP2.c and AP2.d in Appendix 2. The TSA is executed when the current schedule (CT) is earlier than the previous schedule (PVT). Basically TSA moves CR and CT backwards until CT > PVT. This is done by moving the pointer backwards once by TSA and then running the envelop process to reschedule the new police rate. This is repeated until CT > PVT. The TSA, after moving the pointer one step backwards does not clear the NEXT store schedule and thus leaves behind some scheduled police rates. This causes ERROR2. See table 5.3.1 under the column NS_O_C0. While moving tER14+τ3 backwards it has not cleared the schedule. The following example gives a clear picture of this error.

<table>
<thead>
<tr>
<th>N</th>
<th>ER</th>
<th>DEL-PR</th>
<th>NEW-PR</th>
<th>PR-SCHED</th>
<th>PACR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PR10</td>
<td></td>
<td></td>
<td>tER11+τ2→0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>PR14</td>
<td></td>
<td></td>
<td>tER14+τ3→0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ER15</td>
<td>PR14</td>
<td>PR15</td>
<td>tER15+τ3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ER16</td>
<td>PR14</td>
<td>PR14</td>
<td>tER14+τ3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>PR16</td>
<td></td>
<td></td>
<td>tER16+τ3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>PR6</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PR6</td>
<td></td>
<td></td>
<td>0 / tER7+τ2→0</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>PR6/7</td>
<td></td>
<td></td>
<td>0 / tER8+τ2→0</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>PR7/8</td>
<td></td>
<td></td>
<td>0 / tER9+τ2→0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>PR8/9</td>
<td></td>
<td></td>
<td>0 / tER10+τ2→0</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.3.2: N-Store police rate table built by ‘Nothing scheduled’ routine and ‘Something scheduled’ routine of envelop process after the adjustment done in table 3.3.4d that introduced Error2
The tables 5.3.2 and 5.3.2a, which are the extensions to the tables 3.3.4d and 3.3.4e respectively that include additional rates PR15 and PR16 and their schedules. After the TSA adjustment the pointer shows at store 10 (refer table 5.3.2). Consider the arrival of ER15, which is an increase in rate over PR14. After the arrival of ER15 the pointer increments and store 11 becomes the CURRENT store. Since store 11 is unscheduled as shown below, it goes through

```
11. PR14 | 0
```

‘Nothing Scheduled’ routine and the new police rate PR15 is scheduled as tER15⁺ᵣ₃ and store them in CURRENT store (store 11) as shown below.

```
11. PR14/15 | 0/tER15⁺ᵣ₃
```

Consider the arrival of ER16, which is an increase in rate over PR15. After arrival of ER16 the pointer increments and store 12 becomes the CURRENT store. Since the store 12 is already scheduled it goes through

```
12. PR14 | tER14⁺ᵣ₃
```

‘Something Scheduled’ routine and the new police rate PR16 is scheduled as tER16⁺ᵣ₃ and stored in NEXT store (because tER14⁺ᵣ₃ is < tER16⁺ᵣ₃) as shown below.

---

<table>
<thead>
<tr>
<th>Cyclic N Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Police rate</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>1. PR0</td>
</tr>
<tr>
<td>2. PR1</td>
</tr>
<tr>
<td>3. PR2</td>
</tr>
<tr>
<td>4. PR3</td>
</tr>
<tr>
<td>5. PR6</td>
</tr>
<tr>
<td>6. PR7</td>
</tr>
<tr>
<td>7. PR8</td>
</tr>
<tr>
<td>8. PR9</td>
</tr>
<tr>
<td>9. PR10</td>
</tr>
<tr>
<td>10. PR14</td>
</tr>
<tr>
<td>11. PR14/15</td>
</tr>
<tr>
<td>12. PR14</td>
</tr>
<tr>
<td>13. PR16</td>
</tr>
</tbody>
</table>

Table 5.3.2a: Police rates and their schedules determined by Cyclic N Store
Thus the ERROR2 has made it possible for PR14 which is lower than PR15
and PR16 to come in between PR15 and PR16 in the stores list. When PR14
is enforced it causes non-conformance to well behaved sources transmitting
cells at a rate ER15.

The two corrections, CORRECTION 1 and CORRECTION 2, for the
respective errors ERROR 1 and ERROR 2 are described in the next section.

5.3.1.3 CORRECTION 1

ERROR1 has been found to occur during the ‘Nothing scheduled’ routine.
The algorithm never stores the feedback rate (PACR) in CURRENT store.
Instead it stores PACR in the NEXT store and its schedule in the CURRENT
store. The correction for this is to store PACR both in the NEXT and
CURRENT stores and its schedule in the CURRENT store. For this the
algorithm should remove the condition NT <> 0. This is illustrated in the
flow chart AP2.c and corresponding pseudo code in Appendix 2. In fact the
author of the N_Store algorithm has not included the condition NT <> 0 in
the C code published in his previous paper [11].

Refer to table 5.3.1 for the police rates obtained after the CORRECTION 1.
The rates PR1 to PR9 are now scheduled from the respective feedback
arrival times tER1 to tER9.

5.3.1.4 CORRECTION 2

ERROR 2 is due to TSA process in which the NEXT store is not
unscheduled after moving the pointer backwards once, leading to wrong
placement of rates and their schedules in the stores. This can be corrected by
including an additional step NT=0 after the routine moves pointer backward once. This is illustrated in the flow chart in fig. AP2.d in Appendix 2 and corresponding pseudo code in Appendix 2.2. Refer to table 5.3.1 for the police rates obtained after the CORRECTION 1 and 2. The rates PR1 to PR9 are now scheduled from the respective feedback arrival times tER1 to tER9. The schedules of additional PR13 and PR14 are now cleared as shown below.

<table>
<thead>
<tr>
<th></th>
<th>With ERROR2</th>
<th>With CORRECTION 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR14</td>
<td>tER13 + τ3</td>
<td>PR14</td>
</tr>
<tr>
<td>PR13</td>
<td>tER13 + τ3</td>
<td>PR13</td>
</tr>
</tbody>
</table>

After completing CORRECTION 2, ERROR 2 will be eliminated and graphs for NS_O_C2 in fig. 5.3.1 to 5.3.3 show smooth and meaningful curves. Refer to fig 5.3.4 for envelopes of ER and PR after the correction 1 and 2. The gap A and B of the N_Store algorithm with correction 1 and 2 are shown in fig 4.2.1e.

![POLICE RATE ENVELOPE](image)

Fig. 5.3.4

*Cyclic N_Store with error corrections 1 and 2- Police rate envelope (τ₂ = 5 ms and τ₃ = 1 ms)*
5.3.2 Performance results

Figures 5.3.2.1 to 5.3.2.27 show the graphs of percentage of tagged cells, excess cells and police rates obtained for three environments LAN, MAN and WAN and three methods of RTD variations for each environment, in total 27 graphs. Each graph contains the results for Algorithm A, Algorithm B (2_Store) and N_Store. The N_Store results are referred to in the graph as the algorithm N_Store_OC2, which is the N_Store algorithm corrected for 2 errors, ERROR 1 and ERROR 2. Analysis of results is carried out on each type of environment, LAN, MAN and WAN, separately. These provide a clear picture of the magnitude of the impact of each algorithm on the user and the network.

5.3.2.1 LAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for the LAN environment with RTD being varied in 3 ways. They are: 1. Random variation, 2. Random variation with ramping and 3. Variation in accordance with VBR traffic variation.

Figures 5.3.2.1 to 5.3.2.3 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Considering Algorithm A as a reference model, the results show zero percentage tagged cells, a maximum of 15% excess cells and 72% of police rates. The percentages of tagged and excess cells are calculated over the total transmitted cells during the simulation period. The percentage of policed rates are calculated over the total explicit rates (ERs) received by the policer. The results for Algorithm B (2_Store) show almost the same figure as that of Algorithm A. This is mainly because of the smaller values of $\tau_2$ and $\tau_3$ in which very few explicit rates are generated within $\tau_2-\tau_3$. Many times it could be only one. As such, all the ERs generated are taken as police rates. That is why in fig. 5.3.2.3 it
can be seen that the police rates generated are 100% up to a value of $\tau_2=0.4$ milli second and above 70% from $\tau_2=0.4$ to 1 milli second. It is the same reason why the 2_Store being a flat rate policer has the same results as that of Algorithm A. However the results for N_Store_OC2 algorithm are different. The maximum percentage of tagged cells produced by the algorithm is 2.2% for a value of $\tau_2=1$ milli second. This is against the conformance definition whose aim is to make the policer loss less. However, the reason for the tagged cells being generated is analysed in chapter 5. The percentage of police rates has increased by 10% over Algorithm A and hence the percentage of excess cells reduced by 2%. This improvement is obvious as the N_Store algorithm produces more police rates.

The figures 5.3.2.4 to 5.3.2.6 show the results for randomly varied RTD with ramping of increases and decreases. The results are almost the same as those obtained without ramping the RTD variations. Similarly the figures 5.3.2.7 to 5.3.2.9 show the results for RTD being varied in accordance with the variation of VBR traffic. The pattern of results is same but the values are slightly different.

Another point to be noted is the result for a value of $\tau_2=\tau_3=0.1$ milli sec. All the 3 algorithms produce the same result. This is obvious because $\tau_3<$ RTD $<\tau_2$ (RTD is varied between $\tau_2$ and $\tau_3$) and hence RTD $=\tau_2=\tau_3$. That means there is no variation in RTD and it is a fixed propagation delay. The percentage of tagged and excess cells produced is zero and percentage of police rates is 100. This also verifies that the simulation correctness.

From the above results it can be concluded that for the LAN environment, the impact of the poor performance of a policer is minimal due to small variations in RTD. When the delay between $\tau_3$ and $\tau_2$ increases, then the percentage police rates reduce and hence the percentage excess cells and tagged cells become more significant.
Performance of Policing in terms of % Tagged Cells
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

![Graph 5.3.2.1](image1)

Performance of Policing in terms of % Excess Cells
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

![Graph 5.3.2.2](image2)

Performance of Policing in terms of % Police Rates
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

![Graph 5.3.2.3](image3)
Performance of Policing in terms of % Tagged Cells
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 5.3.2.4

Performance of Policing in terms of % Excess Cells
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 5.3.2.5

Performance of Policing in terms of % Police rates
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 5.3.2.6
The ABR policer Simulation

Performance of Policing in terms of % Tagged Cells
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Algorithm A
Algorithm B
Algorithm NS_OC2

Fig. 5.3.2.7

Performance of Policing in terms of % Excess Cells
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Algorithm A
Algorithm B
Algorithm NS_OC2

Fig. 5.3.2.8

Performance of Policing in terms of % Police rates
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Algorithm A
Algorithm B
Algorithm NS_OC2

Fig. 5.3.2.9
5.3.2.2 MAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for MAN environment with RTD being varied in 3 ways. They are; 1. random variation, 2. random variation with ramping and 3. variation in accordance with VBR traffic variation.

Figures 5.3.2.10 to 5.3.2.12 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Figure 5.3.2.10 shows that the percentage tagged cells produced by Algorithm A and B (2_Store) are zero, whereas the N_Store_OC2 produces a maximum of 6% of tagged cells for $\tau_2 = 5$ milli second. The percentage of Tagged cells increases quickly to 5.2%, with $\tau_2$ being increased from 1 milli sec to 1.8 milli sec., and then increases very slowly to a maximum value of 6% at $\tau_2 = 5$ milli sec. This shows that large increases in $\tau_2$ do not produce significantly more tagged cells. The reason is that the main contributions for the tagged cells are from region B of the figure 4.2.1. As discussed in section 4.2.1, the gap B is influenced by the number of rate changes rather than the delay $\tau_2$. The different explicit rates produced by the ERICA switch repeats often. As such, the number of rate changes accumulated over larger values of $\tau_2-\tau_3$, will not vary significantly. Hence the percentage tagged cells does not show a significant increase beyond 1.8 milli sec.

Figure 5.3.2.11 shows that 2_Store Algorithm produces a higher percentage of excess cells and a lower percentage of police rates which is obvious as the 2_Store policer is biased towards higher rates and practically a flat rate policer. It can produce excess cells to a maximum of 35% compared to 32% in Algorithm A and 27% in the N_Store algorithm for $\tau_2=5$milli sec. The N_Store algorithm produces a minimum of 52% of police rates compared to 32% in Algorithm A and 20% in Algorithm B. Even at RTD=$\tau_2=\tau_3=1$milli second the Algorithm B (2_Store) produces only 62% police rates instead of
100% with the remainder. This is because the police rates are overwritten, thus some of them are lost.

The figures 5.3.2.13 to 5.3.2.15 show the results for randomly varied RTD with ramping of increases and decreases and figures 5.3.2.16 to 5.3.2.18 show the results for RTD being varied in accordance with the variation of VBR traffic. The pattern of results is almost the same with slight differences in the values.

From the above results, it can be concluded that for the MAN environment, the impact of poor performance of a policer is quite substantial. Unlike the LAN environment, the algorithm B produces more excess cells and the N_Store algorithm produces more (almost 3 times) tagged cells compared to Algorithm A.
The ABR policer Simulation

Performance of Policing in terms of % Excess Cells
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 5.3.2.10

Performance of Policing in terms of % Tagged Cells
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 5.3.2.11

Performance of Policing in terms of % Police rates
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 5.3.2.12
The ABR policer Simulation

**Performance of Policing in terms of % Tagged Cells**

In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

```
% of Tagged Cells generated when cells are transmitted by source at feedback rate
```

![Graph showing performance of policing in terms of % Tagged Cells](image)

**Fig. 5.3.2.14**

**Performance of Policing in terms of % Excess Cells**

In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

```
% of Excess Cells generated when cells are transmitted by source at Police rate
```

![Graph showing performance of policing in terms of % Excess Cells](image)

**Fig. 5.3.2.14**

**Performance of Policing in terms of % Police rates**

In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

```
% of Police rates generated over feedback RTD cells
```

![Graph showing performance of policing in terms of % Police rates](image)

**Fig. 5.3.2.15**
Performance of Policing in terms of % Tagged Cells
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 5.3.2.16

Performance of Policing in terms of % Excess Cells
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 5.3.2.17

Performance of Policing in terms of % Police Rates
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 5.3.2.18
5.3.2.3 WAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for WAN environment with RTD being varied in 3 ways. They are; 1. random variation, 2. random variation with ramping and 3. variation in accordance with VBR traffic variation.

Figures 5.3.2.19 to 5.3.2.21 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Figure 5.3.2.19 shows that the percentage tagged cells produced by Algorithm A and B (2_Store) are zero, whereas the N_Store_OC2 produces a maximum of 9.8% of tagged cells. The curve behaviour is similar to that explained in section 5.3.2.2.

Figure 5.3.2.20 shows that Algorithm B produces a higher percentage of excess cells and a lower percentage of police rates, which is obvious as the 2_Store policer is biased towards higher rates and is practically a flat rate policer. It produces excess cells to a maximum of 48% compared to 38% in Algorithm A and 32% in the N_Store algorithm for \( \tau_2=15 \text{milli seconds} \). The N_Store algorithm produces a minimum of 26% of police rates compared to 18% in Algorithm A and 6% in Algorithm B. At RTD=\( \tau_2=\tau_3=5 \text{ milli seconds} \), only Algorithm A produces 100% police rates without loosing any feedback rates whereas the Algorithm B (2_Store) produces only 18% and N_Store 76%. Figures 5.3.2.22 to 5.3.2.24 show the results for randomly varied RTD with ramping of increases and decreases and figures 5.3.2.25 to 5.3.2.27 show the results for RTD being varied in accordance with the variation of VBR traffic. The patterns of results are almost same with only slight differences in the values.

From the above results, it can be concluded that for the WAN environment, the impact of poor performance of a policer is quite significant. Unlike the LAN or MAN environment, the algorithm B produces a higher percentage of excess cells and a lower percentage of police rates than Algorithm A,
proving to be unsuitable as a policer, particularly for WAN. The tagged cells for the N_Store algorithm has not much increased from that of the MAN environment as we mentioned above, due to a lesser influence on larger delays.
The ABR policer Simulation

Performance of Policing in terms of % Tagged Cells
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Algorithm A
Algorithm B
Algorithm NS OC2

Fig. 5.3.2.19

Performance of Policing in terms of % Excess Cells
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Algorithm A
Algorithm B
Algorithm NS OC2

Fig. 5.3.2.20

Performance of Policing in terms of % Police Rates
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Algorithm A
Algorithm B
Algorithm NS OC2

Fig. 5.3.2.21
The ABR policer Simulation

Fig. 5.3.2.22
Performance of Policing in terms of % Tagged Cells
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 5.3.2.23
Performance of Policing in terms of % Excess Cells
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 5.3.2.24
Performance of Policing in terms of % Police rates
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)
The ABR policer Simulation

### Performance of Policing in terms of % Tagged Cells
In WAN Environment with RTD varying in accordance with VBR traffic variation (\(\tau_3 = 5000\) Micro secs. and \(\tau_2 =\) Varied from 5000 to 10000 Micro secs.)

![Graph](attachment:graph1.png)

**Fig. 5.3.2.25**

### Performance of Policing in terms of % Excess Cells
In WAN Environment with RTD varying in accordance with VBR traffic variation (\(\tau_3 = 5000\) Micro secs. and \(\tau_2 =\) Varied from 5000 to 10000 Micro secs.)

![Graph](attachment:graph2.png)

**Fig. 5.3.2.26**

### Performance of Policing in terms of % Police rates
In WAN Environment with RTD varying in accordance with VBR traffic variation (\(\tau_3 = 5000\) Micro secs. and \(\tau_2 =\) Varied from 5000 to 10000 Micro secs.)

![Graph](attachment:graph3.png)

**Fig. 5.3.2.27**
The result supports the performance analysis that was made in section 5.2, indicating that the N_Store algorithm is not loss less, due to some inherent error. The 6% to 10% tagged cells that may be subjected to loss during congestion are significant for a well-behaved user. This adversely affects the credibility of a network committed to providing a guaranteed QoS for a compliant connection. Thus the N-Store algorithm, which has been recommended to the ATM Forum for the practical implementation of Algorithm A [3], does not perform to the same level as Algorithm A in all the three LAN, MAN and WAN environments. It could not be possible to come to this conclusion without the new approach in the performance measurement presented in this thesis.

It has been seen that all the algorithms produce excess cells. This is due to the conformance definition in which an increase in Explicit Rate is scheduled as a police rate at the minimum RTD=τ₃ and at the maximum RTD=τ₂ for a decrease in ER. This allows a big gap between the actual feedback rate and the police rate. This is inevitable because of unpredictable variations in feedback delay. The main objective in defining the conformance definition this way is to ensure that no cell-loss (non-conformance) occurs to a compliant connection because of the policing under such variable RTD conditions.

Allowing excess traffic to flow into the network without a quantitative knowledge may degrade the network service due to lack of resource. If the quantity is known to a fair value then the network operator can prepare in advance with resources to face worst case situations. Besides, allowing algorithms, no matter how simple they are, to produce excess cells more than a normal allowance (Algorithm A may be taken as the bench mark) may affect the network performance due to lack of resources. In particular, it may affect other users sharing the common link. Considering all these points, the 2_Store policer, even though very simple to implement, appears a poor choice for policing. The N_Store policer, which is claimed to be the
equivalent to Algorithm A, appears not to be so with respect to its performance. It takes 10% excess bandwidth more than that of Algorithm A and produces 6% tagged cells.

5.4 Conclusions:

This chapter has examined the performance of three policer algorithms in terms of percentage cell-loss and excess cells. All the algorithms except N_Store are lossless. Even though, the cyclic N_Store algorithm is more accurate in producing more police rates, it produces a significant percentage of tagged cells for compliant connections, potential excess cells from malicious users, and is still complex to implement. The quantitative measurement of cell-loss and excess cells revealed the real performance of the policers. Until the measurements in this thesis were presented, the N_Store algorithm was believed to be equivalent to Algorithm A. After measurement however it showed a poor performance compared to Algorithm A. These factors motivated the improvement of the existing N_store algorithm to make it lossless and simple to implement, this is presented in chapter 5.

The ABR policer performance investigation has focussed on the cyclic N_Store algorithm, which is claimed to be equivalent to Algorithm A and simple to implement. Section 4.4 revealed the constraints in the choice of $\tau_2$ and $\tau_3$. The section 4.6 examined the effect of choice of $\tau_2$ on the performance of the policer. Our results have shown that:

- The excess cells increase with an increase in $\tau_2$. This is because of the increase in the gap between the police rate envelope and feedback explicit rate envelope. The conformance definition, which is biased towards higher rate policing, is responsible for this behaviour.
- The percentage of excess cells for the 2_Store algorithm is more than Algorithm A and the N_Store algorithm. The algorithm at any point of time memorises only 2 feedback rates, so those higher rates are policed for a longer time.

- The percentage of excess cells for the N_Store algorithm is marginally lower than for Algorithm A. This reveals the error in the N_Store algorithm. Given that Algorithm A memorises all the feedback rates, the N_Store should produce fewer excess cells than Algorithm A.

- The percentage of tagged cells in all the algorithms except in the N_Store algorithm is found to be zero. Due to some inherent error in N_Store algorithm, it produced tagged cells.

- The percentage of tagged cells is found to increase more with source rate changes than with RTD.

The results show that the Algorithm A seems to be perfect as a policer as it produces no cell-loss and it allows only the expected excess cells. However, it is very complex to implement. The 2_Store algorithm is not good as a policer due to its inability to generate a greater number of police rates and the possibility of allowing more excess cells than that of Algorithm A. However, it is loss less and simple to implement. Although the N_Store algorithm does not produce any extra excess cells, it does incur cell-loss for well-behaved users, which is against the aim of the conformance definition.

From the above performance measurements, it appear that another algorithm is needed which should be simple to implement, should not produce more excess cells than Algorithm A and should be loss less. Such an improved N_Store algorithm is proposed in chapter 6.
6. Improved N_Store Algorithm and its Performance Review

6.1 Introduction

In this chapter improvements to the N-Store algorithm are proposed to overcome the inefficiencies described in chapter 5. It begins by reviewing the police rate envelope (fig 6.2.1) and analysing the cause for the inaccuracy, which produces a high percentage of tagged cells. A new Rate Sequence Check (RSC) algorithm is introduced which, when added to the original N_Store algorithm, will eliminate the tagged cells. This is followed by showing the 'Improved N_Store' is loss less. Finally, the simplified N_Store_I algorithm is introduced and is shown that the algorithm is equivalent to Algorithm A and is also loss less.

Section 6.2 reviews the police rate envelope formed by the N_Store Algorithm and analyses it to find the reasons for inaccuracy. In section 6.3 the new Rate Sequence Check (RSC) algorithm is introduced to show how the performance is improved by adding this to the main algorithm. The simplification of the N_Store algorithm is shown through merging several blocks into one block. Section 6.4 shows the performance of the improved algorithm and the simplified algorithm N_Store_I. The conclusion are given in section 6.5.
6.2 Inaccuracy in the N_Store Algorithm

It has been discussed in previous chapters how the police rate envelope of the N_Store algorithm produces two types of gaps A and B (ref fig.4.2.1a to 4.2.1c and 6.2.1). Within gap B, higher rates are policed by the lower police rates. For example, in fig.6.2.1 PR4 is within gap B. The forward cells adapting the feedback rate ER6 may arrive at the policer anywhere between \( t_{ER6} + \tau_3 \) and \( t_{ER6} + \tau_2 \), depending upon the RTD. The policer enforces PR4 at \( t_{ER4} + \tau_2 \), which definitely falls within the period \( t_{ER6} + \tau_3 \) and \( t_{ER6} + \tau_2 \). If forward cells adopting ER6 arrive at the policer when \( RTD > \tau_2 - [(t_{ER6} + \tau_2) - (t_{ER4} + \tau_2)] \) then the cells become non-conforming due to the lower police rate, thus creating more tagged cells within gap B. Similarly within gap A, lower rates are policed by much higher rates than normal. For example, cells arriving at rates ER4 to ER7 are policed by a much higher rate, PR3.

ERROR 1 and ERROR 2 have been discussed in section 5.3. The error corrections have eliminated the majority of gap A and B. Hence all the graphs in fig. 5.3.2.1 to 5.3.2.27 show less excess cells and more tagged cells than those of Algorithm A. Although it could be expected that the elimination of gap A should have made the excess cells equal to those of Algorithm A, the results do not show this. The results show less excess cells than that of Algorithm A. This is because of gap B, which has reduced the total gap between police rate envelope of Algorithm A and the feedback explicit rate envelope. Hence there are fewer excess cells. Elimination of gap B should obviously reduce the tagged cells to zero and increase the excess cells to be equal to those of Algorithm A, thus making N_Store Algorithm equivalent to Algorithm A.

Further analysis found that the cause of gap B is due to inaccuracy in scheduling a higher rate after several decreases in rates. The inaccuracy
occurs when there is an increase in the explicit rate over the previous decreases in rates but a decrease over the enforced police rate. Consider ER6 and ER10 table 6.2.1 under NS_O_C2. It is illustrated in figs.6.2.1 and 6.2.2, that the ER6 and ER10 are higher than their respective previous rates ER5 and ER9 but less than the police rates PR3 and hence the N_Store algorithm schedules PR6 and PR10 after \( \tau_2 \). PR4 and PR5 are enforced as police rates at \( t_{ER4} + \tau_2 \) and \( t_{ER5} + \tau_2 \) earlier than PR6 which is enforced at \( t_{ER6} + \tau_2 \). As such, the cells transmitted by the source at a rate ER6 may be received at policer anywhere between \( t_{ER6} + \tau_3 \) to \( t_{ER6} + \tau_2 \). Refer to fig. 6.2.1 in which the two ER envelopes are shown with one after \( \tau_3 \) and the other after \( \tau_2 \). Depending upon the RTD value the source cells transmitted at a rate ER6 may fall within the gap B shown in fig 6.2.1 and cells get non conformed. This inaccuracy which creates gap B is due to higher rates being scheduled latter than lower rates. The next section proposes the modification to overcome this inaccuracy.
6.3 Rate sequence check (RSC) and accurate N_Store Algorithm

It was found in the previous section that inaccuracies occur because the algorithm schedules higher rates after the arrival of previous lower rates. Refer to stores 6 and 9 for NS_O_C2 algorithm in table 6.2.1. If the algorithm replaces PR4 and PR5 with PR6 and PR8, PR9 with PR10 then, the gap B closes. With this, the police rates envelop exactly matches with that of Algorithm A. Correcting just the previous rate is not enough. There could be a chain of decreases in rates previous to an increase in rates. In such case, all the possible decreases in rates need to be overwritten by the increased rate. This process is referred as Rate sequence check (RSC).

Refer to the flow chart in AP2.g and the pseudo code in Appendix 2 which illustrates the operation of RSC. Once the increase in the rate over the previous decrease is scheduled in the CURRENT store, then the routine checks whether the rate in the CURRENT store is greater than the rate in PREVIOUS store. If it is, then it checks whether the PREVIOUS rate
schedule is greater than or equal to (arrival time of CURRENT rate + \( t_3 \)). The schedule is tested mainly to ensure that the rates with schedules earlier than the minimum enforcing time of the higher rate are not overwritten. Once this is ensured then the pointer is moved backward once. The envelope process particularly the ‘Schedule new rate first’ (refer to the flow chart in AP2.b in Appendix 2) stores the higher rate in the CURRENT store. However before scheduling the CURRENT store, it checks the setting of the rate sequence check flag (rscf). If the flag is set, then it does not change the CURRENT schedule. That means it keeps the original PREVIOUS rate schedule. This is very important, because the higher rate needs to be scheduled at the schedule of the lower rate, which it replaced. After this, it resets the flag. At the end of the routine, the NEXT store is unscheduled.

The above routine is repeated until one of the two conditions is met. That is either the PREVIOUS rate should be higher than the CURRENT rate (PVR>CR) or the PREVIOUS schedule earlier than CURRENT schedule (PVT < CT). After the inclusion of RSC as an improvement to overcome the inaccuracy, the police rate envelope exactly matches that of Algorithm A (refer to Stores 5 and 7 for NS_O_M in table 6.2.1). The store 5 contains PR6 scheduled at \( t_{ER4+\tau_2} \) and store 7 contains PR10 scheduled at \( t_{ER8+\tau_2} \). This totally eliminates gap B.

Thus with the addition of RSC the N_Store algorithm has become loss less and now can be claimed to be equivalent to Algorithm A in the performance. However the modified algorithm NS_O_M is to be verified through simulation results. The next section gives these results, which are comparable to Algorithm A.
6.4 Results

6.4.1 Performance Results

Figures 6.4.1 to 6.4.27 show the graphs of the percentage of tagged cells, excess cells and police rates obtained for three environments LAN, MAN and WAN and three methods of RTD variations for each environment in total 27 graphs. Each graph contains the results for Algorithm A and modified N_Store algorithm NS_O_M. Analysis of the results are carried out on each type of environment, LAN, MAN and WAN, separately.

6.4.1.1 LAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for LAN environment with RTD being varied in 3 ways. They are; 1. random variation, 2. random variation with ramping and 3. variation in accordance with VBR traffic variation.

Figures 6.4.1 to 6.4.3 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Considering Algorithm A as a perfect model, the results show zero percentage tagged cells, a maximum of 15% excess cells and 72% of police rates. The curves for NS_O_M match the Algorithm A and hence these two algorithms are equivalent. Figs 6.4.4 to 6.4.6 show the results obtained for randomly varied RTD with ramping of increases and decreases and figs 6.4.7 to 6.4.9 show the results obtained for RTD varied in accordance with VBR traffic. The results are same as those for randomly varied RTD but with slight variations.
Improved N_Store Algorithm and its performance review

Performance of Policing in terms of % Tagged Cells
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.1

Performance of Policing in terms of % Excess Cells
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.2

Performance of Policing in terms of % Policed rates
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.3
Performance of Policing in terms of % Tagged Cells
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.4

Performance of Policing in terms of % Excess Cells
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.5

Performance of Policing in terms of % Police rates
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.6
Improved N_Store Algorithm and its performance review

Fig. 6.4.7

Performance of Policing in terms of % Tagged Cells
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.8

Performance of Policing in terms of % Excess Cells
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.4.9

Performance of Policing in terms of % Police rates
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)
6.4.1.2 MAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for MAN environment with RTD being varied in 3 ways. They are; 1. random variation, 2. random variation with ramping and 3. variation in accordance with VBR traffic variation.

Figures 6.4.10 to 6.4.12 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Considering Algorithm A as a perfect model, the results show zero percentage tagged cells, a maximum of 33% excess cells and a minimum of 32% of police rates. For the MAN environment, the maximum percentage of excess cells is increased by 18% and a minimum percentage of the police rates is decreased by 40% compared to that of LAN environment. This is due to an increase in delay by 5 times. The curves for NS_O_M match with those of Algorithm A and hence the two algorithms are equivalent. Figures 6.4.13 to 6.4.15 show the results obtained for randomly varied RTD with ramping of increases and decreases and figures 6.4.16 to 6.4.18 show the results obtained for RTD varied in accordance with VBR traffic. The results are the same as that for randomly varied RTD with slight variations.
Fig. 6.4.10

Performance of Policing in terms of % Excess Cells
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.11

Performance of Policing in terms of % Police rates
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.12
Performance of Policing in terms of % Tagged Cells
In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.13

Performance of Policing in terms of % Excess Cells
In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.14

Performance of Policing in terms of % Police rates
In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.15
Performance of Policing in terms of % Tagged Cells
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.16

Performance of Policing in terms of % Excess Cells
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.17

Performance of Policing in terms of % Police Rates
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.4.18
6.4.1.3 WAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for WAN environment with RTD in 3 ways. They are; 1. random variation, 2. random variation with ramping and 3. variation in accordance with VBR traffic variation.

Figures 6.4.19 to 6.4.21 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Considering Algorithm A as a perfect model, the results show zero percentage tagged cells, a maximum of 43% excess cells and a minimum of 18% of police rates. For the WAN environment, the maximum percentage of excess cells is increased by 29% and a minimum percentage of the police rates is decreased by 54% compared to that of LAN environment. When compared to MAN environment the maximum percentage of excess cells is increased by 11% and minimum percentage of Police rates is decreased by 14%. This is due to an increase in delay by 15 times compared LAN and 3 times compared to MAN. The curves for NS_O_M match with those of Algorithm A with practically zero percentage of tagged cells and hence the two algorithms are equivalent.

Figures 6.4.22 to 6.4.24 show the results obtained for randomly varied RTD with ramping of increases and decreases and figures 6.4.25 to 6.4.27 show the results obtained for RTD varied in accordance with VBR traffic. The results are the same as those for randomly varied RTD but with slight variations.
Improved N_Store Algorithm and its performance review

Performance of Policing in terms of % Tagged Cells
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs, and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.4.19

Performance of Policing in terms of % Excess Cells
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs, and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.4.20

Performance of Policing in terms of % Police Rates
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs, and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.4.21
Performance of Policing in terms of % Tagged Cells
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

![Graph of % Tagged Cells](image)

Fig. 6.4.22

Performance of Policing in terms of % Excess Cells
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

![Graph of % Excess Cells](image)

Fig. 6.4.23

Performance of Policing in terms of % Police rates
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

![Graph of % Police rates](image)

Fig. 6.4.24
Performance of Policing in terms of % Tagged Cells
In WAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

![Graph showing performance of policing in terms of % Tagged Cells.]

Fig. 6.4.25

Performance of Policing in terms of % Excess Cells
In WAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

![Graph showing performance of policing in terms of % Excess Cells.]

Fig. 6.4.26

Performance of Policing in terms of % Police rates
In WAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

![Graph showing performance of policing in terms of % Police rates.]

Fig. 6.4.27
6.5 Simplified N_Store_I Algorithm:

The original [3] N_Store algorithm, as given in Appendix 2, is complex. When compared to the 2_Store algorithm, the pseudo code for the N_Store algorithm is 5 to 6 times longer. The complexity arises from the envelope process, which covers 70% of the algorithm. An effort has been made to reduce the original N_Store algorithm to a simplified N_Store_I algorithm. This is basically the original N_Store + corrections for Error1 and Error2 + RSC + reduced envelope process.

A flowchart of the envelope process of the original N_Store algorithm is given in fig.5.5.1. Its purpose is to evaluate a police rate and its schedule under the following 3 conditions by which an explicit rate is received through backward RM cell.

1. The received ER is an increase over the PREVIOUS rate and the running police rate.
2. The received ER is an increase over the PREVIOUS rate but a decrease over the running police rate.
3. The received ER is a decrease over the PREVIOUS rate.

The N_Store algorithm evaluates the police rate and its schedule for the above conditions as follows.

Condition 1: The police rate as ER and the schedule as Arrival Time + τ₃
Condition 2. The police rate as ER and the schedule as Arrival Time + τ₂
Condition 3. The police rate as ER and the schedule as Arrival Time + τ₂

The Arrival Time is the time at which the backward RM cell is received.

The rates and schedules are stored either in CURRENT or NEXT store depending upon the available schedule in the CURRENT store. They are as follows.
Condition 1. The schedule is stored in the ‘NEXT store’ if the CURRENT store is already scheduled earlier than Arrival Time + τ₃, otherwise it is stored in the CURRENT store.

Condition 2. The schedule is always stored in CURRENT store.

Condition 3. The schedule is stored in NEXT store if CURRENT store is already scheduled, otherwise it is stored in CURRENT store.

The above storage details are shown in figure 6.5.1 inside the boxes A to G. The above analysis shows that the algorithm needs two routines. One for evaluating the police rate and its schedule and the second one for storing them either in the CURRENT or the NEXT store.

In the algorithm, the rate and schedule evaluation routine are straightforward. This routine is repeated twice in the algorithm, once for the ‘Something scheduled routine’ and the second time for ‘Nothing scheduled routine’. This can be avoided by retaining it in only one routine i.e. ‘Nothing scheduled routine’.

The question of storage must be considered which complicates the algorithm. This is the one, which has complicated the algorithm. The algorithm has to check various conditions and repeat logics and routines to decide where to store the rate and schedule. This can be avoided by intelligently merging the several functions within one function. This is described as follows.

Consider the routine (shaded portion in fig. 6.5.1) under ‘nothing scheduled’, which is the only one retained for envelope process. In this routine, the new police rate and its schedule is always stored in the CURRENT store (refer figure AP2.c in Appendix 2). The NEXT store contains only the rate with an empty schedule. Every time a RM cell arrives with a new ER, the routine evaluates a new police rate and its schedule and stores it in the CURRENT store and thus builds up a table of N stores. After each CURRENT store and schedule is built up, it is subjected to a RSC and Time Sequence Adjustment...
Improved N_Store Algorithm and its performance review

(TSA). RSC takes care of scheduling the increase in rate over the previous rate decreases whereas TSA takes care of scheduling the increase in rate over running police rate and a previous rate decrease. Both routines basically place the higher rates front in the queue in N store for first in, first out delivery.

The table 6.5.1 shows the rates and their schedules obtained from the new N_Store_I algorithm. In store 5 and 7, the rates are subjected to RSC and thus the higher rates PR6 and PR10 are scheduled at tER4+τ2 and tER8+τ2 (schedules of previous decreases in rates ER4 and ER8) respectively. The time sequence adjustment will simply move the earliest scheduled rate (tER13+τ3) to the right position in N store.

<table>
<thead>
<tr>
<th>ER</th>
<th>N</th>
<th>RATE</th>
<th>SCHEDULE</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ER0</td>
<td>1</td>
<td>PR0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>ER1</td>
<td>2</td>
<td>PR1</td>
<td>tER1+τ1</td>
<td></td>
</tr>
<tr>
<td>ER2</td>
<td>3</td>
<td>PR2</td>
<td>tER2+τ1</td>
<td></td>
</tr>
<tr>
<td>ER3</td>
<td>4</td>
<td>PR3</td>
<td>tER3+τ3</td>
<td></td>
</tr>
<tr>
<td>ER4</td>
<td>5</td>
<td>PR4/5/6</td>
<td>tER4+τ2</td>
<td>RSC is done here</td>
</tr>
<tr>
<td>ER5</td>
<td>6</td>
<td>PR7</td>
<td>tER7+τ2</td>
<td></td>
</tr>
<tr>
<td>ER6</td>
<td>7</td>
<td>PR8/9/10</td>
<td>tER8+τ2</td>
<td>RSC is done here</td>
</tr>
<tr>
<td>ER7</td>
<td>8</td>
<td>PR11/12/13</td>
<td>tER11/12/13+τ3/τ2/τ1</td>
<td>TSA is done here</td>
</tr>
<tr>
<td>ER8</td>
<td>9</td>
<td>PR14</td>
<td>tER14+τ3</td>
<td></td>
</tr>
<tr>
<td>ER9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER13</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6.5.1: Police rates and their schedules determined by N_Store_I

Refer to store boxes A to D under ‘something scheduled routine’ in fig.6.5.1. The store boxes A and B contain rates in the NEXT store. This is done because CURRENT store is already scheduled. In N_Store_I, since CURRENT store is always unscheduled, the routine never stores any rate or schedule in NEXT store. So, the rates are stored in CURRENT store and thus the store boxes G and F take care of stores in boxes A and B. Similarly stores
in boxes C and D are taken care of by E and F respectively through TSA and RSC routines.

Thus the simplified N_Store_I enhances the original N_Store with additional RSC routine. This has greatly simplifies the algorithm and reduced the code size approximately by half. The police rate envelope of the N_Store_I exactly matches Algorithm A. Thus N_Store_I is a simplified and accurate algorithm that is equivalent to Algorithm A. The detailed flow chart and pseudo code is given in Appendix 3.

The performance of the N_Store_I is measured and compared with other algorithms in the next section.
Improved N_Store Algorithm and its performance review

CR: Current Rate, CT: Current rate schedule
NR: Next Rate, NT: Next rate schedule
AT: RM cell arrival time
PVR: Previous Rate, PVT: Previous rate schedule
PACR: Feedback rate ER is stored in PACR
PACRM: Previous maximum PACR
PR: Police rate, TSA: Time sequence adjustment
RSC: Rate sequence check

Fig. 6.5.1. Simplified flow chart of Cyclic N_Store I algorithm

---

End of document. If you have any further questions or need additional assistance, feel free to ask!
6.5.1 Performance results

Figures 6.5.1.1 to 6.5.1.27 show the graphs of the percentage of tagged cells, excess cells and police rates obtained for three environment, LAN, MAN and WAN, and three methods of RTD variations for each environment, in total 27 graphs. Each graph contains the results for Algorithm A and the simplified N_Store_I algorithm. Analysis of results is carried out on each type of environment, LAN, MAN and WAN, separately in subsequent sections.

6.5.1.1 LAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for LAN environment with RTD being varied in 3 ways. They are; 1. random variation, 2. random variation with ramping and 3. variation in accordance with VBR traffic variation.

Figures 6.5.1.1 to 6.5.1.3 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Figures 6.5.4 to 6.5.6 show the results obtained for randomly varied RTD with ramping of increases and decreases and figures 6.5.1.7 to 6.5.1.9 show the results obtained for RTD varied in accordance with VBR traffic. The results are the same as that for randomly varied RTD with slight variations. The curves for N_Store_I match with those of Algorithm A and hence the two results are equivalent.
Performance of Policing in terms of % Tagged Cells
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.1

Performance of Policing in terms of % Excess Cells
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.2

Performance of Policing in terms of % Policed rates
In LAN Environment with randomly varied RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.3
Improved N_Store Algorithm and its performance review

Performance of Policing in terms of % Tagged Cells
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.4

Performance of Policing in terms of % Excess Cells
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.5

Performance of Policing in terms of % Police rates
In LAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.6
Improved N_Store Algorithm and its performance review

Performance of Policing in terms of % Tagged Cells
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.7

Performance of Policing in terms of % Excess Cells
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.8

Performance of Policing in terms of % Police rates
In LAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 100 Micro secs. and Tau2 = Varied from 100 to 1000 Micro secs.)

Fig. 6.5.1.9
6.5.1.2 MAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for MAN environment with RTD being varied in 3 ways. They are; 1. random variation, 2. random variation with ramping and 3. variation in accordance with VBR traffic variation.

Figures 6.5.1.10 to 6.5.1.12 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Figures 6.5.1.13 to 6.5.1.15 show the results obtained for randomly varied RTD with ramping of increases and decreases and figures 6.5.16 to 6.5.18 show the results obtained for RTD varied in accordance with VBR traffic. The results for all 3 methods of RTD have only slight variations. The curves for N_Store_I match with those of Algorithm A and hence the two algorithms are equivalent.
Performance of Policing in terms of % Tagged Cells
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.5.1.10

Performance of Policing in terms of % Excess Cells
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.5.1.11

Performance of Policing in terms of % Police rates
In MAN Environment with randomly varied RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.5.1.12
Performance of Policing in terms of % Tagged Cells
In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.5.1.13

Performance of Policing in terms of % Excess Cells
In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.5.1.14

Performance of Policing in terms of % Police rates
In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

Fig. 6.5.1.15
Performance of Policing in terms of % Tagged Cells
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

![Graph of Performance of Policing in terms of % Tagged Cells](image)

**Fig. 6.5.1.16**

Performance of Policing in terms of % Excess Cells
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

![Graph of Performance of Policing in terms of % Excess Cells](image)

**Fig. 6.5.1.17**

Performance of Policing in terms of % Police Rates
In MAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 1000 Micro secs. and Tau2 = Varied from 1000 to 5000 Micro secs.)

![Graph of Performance of Policing in terms of % Police Rates](image)

**Fig. 6.5.1.18**
6.5.1.3 WAN environment

This section discusses the results obtained in terms of percentage excess cells, tagged cells and police rates for WAN environment with RTD being varied randomly, random with ramping and in accordance with VBR traffic variation.

Figures 6.5.1.19 to 6.5.1.21 show the percentage of tagged cells, excess cells and police rates for randomly varied RTD. Figures 6.5.1.22 to 6.5.1.24 show the results obtained for randomly varied RTD with ramping of increases and decreases and figures 6.5.1.25 to 6.5.1.27 show the results obtained for RTD varied in accordance with VBR traffic. The results for all three methods of RTD are similar with slight variations. The curves for N_Store_I match with those of Algorithm A and hence two algorithms are equivalent.
Improved N_Store Algorithm and its performance review

Performance of Policing in terms of % Tagged Cells
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.19

Performance of Policing in terms of % Excess Cells
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.20

Performance of Policing in terms of % Police Rates
In WAN Environment with randomly varied RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.21
Performance of Policing in terms of % Tagged Cells
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.22

Performance of Policing in terms of % Excess Cells
In MAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.23

Performance of Policing in terms of % Police rates
In WAN Environment with randomly varied (Ramping Up and Down) RTD
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.24
Performance of Policing in terms of % Tagged Cells
In WAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.25

Performance of Policing in terms of % Excess Cells
In WAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.26

Performance of Policing in terms of % Police rates
In WAN Environment with RTD varying in accordance with VBR traffic variation
(Tau3 = 5000 Micro secs. and Tau2 = Varied from 5000 to 15000 Micro secs.)

Fig. 6.5.27
6.6 Conclusions:

The N_Store_I algorithm is a simplified version of the original N_Store. The two errors and one inaccuracy seen in the original algorithm have been rectified in N_Store_I. The N_Store_I performance has greatly improved over the original algorithm by eliminating the tagged cells and improving the excess cells. The tagged cells have been reduced from a maximum of 6 to 10% to practically zero percentage in all LAN, MAN and WAN environments. N_Store_I is essentially equivalent to Algorithm A. N_Store_I is simple to implement and loss less.
7. Conclusions

7.1 Overview

This thesis has considered the performance measurement of the ABR policer. In the process of performance analysis of the results obtained, it has identified errors in the N_Store algorithm. The proposed solution has produced a simplified, accurate N_Store_I algorithm. It has shown, through the performance measurement that the N_Store_I algorithm is equivalent to Algorithm A. The main areas covered in this thesis were:

- Development and analysis of a new method for the performance measurement of the ABR policer related algorithms.

- Development of the performance modelling of the ABR policer located at the User network Interface.

- Performance analysis of the N_Store algorithm, identification of errors as well as corrections to improve the performance.

- Development and performance analysis of a simplified N_Store_I algorithm.

The ATM network multiplexes various traffic types but the majority of which are bursty. The network may become congested if sufficient resources are not reserved and the traffic management functions are not adopted. CAC
and traffic policing are the traffic control functions that are used to avoid the network congestion. The need for policing, their key role in traffic management and the significance of performance measurement in maintaining the QoS guaranteed to the user were described in chapter 2. Having established the importance of policing in ATM networks, the thesis has concentrated on the ABR policing and its performance measurement in subsequent chapters.

The rate based ABR flow control is a traffic control mechanism that avoids congestion in the network. The QoS commitments in the ABR service are bound by a traffic contract with the user. Violation of the contract by any user degrades the QoS commitments to other users who are sharing the common link. Hence a policer is needed to identify such violations and punish non-conforming cells, either by tagging or discarding.

The accuracy of a policer in identifying a conforming cell depends on the conformance definition and the algorithm used to implement it. The ATM Forum has specified the conformance definition framework, therefore these police related algorithms (DGCRA, A, 2_Store and N_Store) have been designed to detect, cell by cell, the conformance of incoming cells from the ABR source. The conformance definition includes two delays, \( \tau_1 \) and \( \tau_2 \). The police rate algorithms designed to take care of the above delay constraints were described in chapter 3.

The accuracy of the chosen algorithm plays a key role in the policing function and traffic management, as described in chapter 2. Many literatures were reviewed regarding Type 1 and Type 2 errors and the accuracy measurement of the GCRA (leaky bucket) algorithm. However, the question of measuring accuracy of ABR policer still remains unanswered. The ABR policer adopts two algorithms. The first algorithm (DGCRA) is for the cell rate conformance check in regard to a reference police rate, and the second algorithm (Algorithm A or B or N_Store) is for generating the police rate
based on the feedback explicit rates (ER) received by backward RM cells. The DGCRA operation is the same as the GCRA, except that the increment $I = 1/PACR$ varies dynamically. Hence, the accuracy of DGCRA can still be defined in terms of Type 1 and Type 2 errors. However, if the police rate $PACR$, and its schedule for enforcing is not accurate enough to meet the requirement of the conformance definition, then inaccurate policing results. Hence, the need for measuring the accuracy of all policer functions. This was introduced in chapter 4 where new approach to the performance measurement of the ABR policer was presented.

This new approach is based on the network criteria for deploying a policer at the UNI. The network expects that the policer not only detects non-conforming but also not detects conforming cell falsely as non-conforming (Type 1 error) or a non-conforming cell as conforming (Type 2 error). The 2 parameters, which quantify these 2 false detection, are respectively, tagged cells and excess cells. It has been shown that the amount of excess cells and the cell-loss can directly measure the effectiveness of a policer. With the theoretical analysis of the algorithm depicted in fig 4.2.1, it was shown that the N_Store algorithm, while meant to implement Algorithm A, in fact, does not. This was proved by quantifying loss of cells and excess cells through a simulation. A new method was introduced, which measures the performance of a policer in terms of percentage tagged cells, percentage of excess cells and percentage of police rates.

An ABR policer simulation model was presented in chapter 4. The model is built to check the conformance of a cell accurately under varying conditions of RTD with all the three algorithms. Results showed that the differences were minimal over a range of RTD variations. This ABR policer model is one of the major contributions of this thesis.

The results showed the following:-
Conclusions

1. Unlike Algorithm A and 2_Store Algorithm B, the results for N_Store algorithm showed errors. On further investigation this was found to be because of two inherent errors (Error1 and Error2) in the algorithm. This was the reason for the creation of gap A as shown in fig. 6.2.1. The errors were fixed in the algorithm NS_O_C2 and the results were found to be comparable to algorithms A and B. It was shown that gap A was eliminated with the correction of the algorithm.

2. The Results of Algorithm A, 2_Store Algorithm B and N_Store NS_O_C2 algorithm were compared and analysed. The results showed that algorithms A and B had zero percentage of tagged cells, whereas it was a maximum of 2%, 6% and 8% for the NS_O_C2 algorithm in LAN, MAN and WAN environment respectively. This was due to gap B as shown in fig. 6.2.1.

3. Compared to Algorithm A, the 2_Store algorithm results showed more excess cells whereas the NS_O_C2 algorithm showed fewer. Similarly, the percentage of police rates for 2_Store were lower than that for Algorithm A whereas it was higher for the NS_O_C2 algorithm. From the results, it can be concluded that the higher the police rates, the fewer the excess cells. The results also showed that for a LAN environment Algorithm A and the 2_Store algorithm B had the same performance.

The analysis of the results obtained through the new approach showed that the performance of the N_Store algorithm is not equal to that of Algorithm A. It has some inherent errors that create the gaps A and B responsible for more tagged cells and excess cells. The results also showed that the 2_Store algorithm is equivalent to algorithm A in a LAN environment. These conclusions can only be made through quantification of excess cells and tagged cells. With this, it is easy to measure the effectiveness of a policer and the network operator can make the best choice of an algorithm. Hence it can be concluded that the performance of a police rate algorithm must be
measured in terms of tagged cells and excess cells and compared with Algorithm A as a benchmark.

However gap B, which was responsible for tagged cells, was analysed in chapter 5 and found to be due to inefficiency in the algorithm, in scheduling a higher rate over the previous decreases in rates. In chapter 5 it is shown that an additional ‘Rate sequence check’ routine has overcome this inefficiency. Adding this routine has shown that the gap B has been eliminated. The results of the completely modified N_Store algorithm (NS_O_M) showed that the algorithm matches the results with Algorithm A in LAN, MAN and WAN environments and all types of RTD variations. The results show that the algorithm is loss less (zero percentage tagged cells) and without any A or B gaps.

Through these error corrections and the additional RSC routine, it has been shown that the performance of the N_Store algorithm now matches that of algorithm A. However, the algorithm is now more complex. The length of the code for the N-Store algorithm is found to be 4 to 5 times longer than that of 2_Store. The most complicated process of implementation was found to be the envelope process. A complete analysis of the envelope process was carried out through the flow chart shown in fig.5.5.1 and after analysis, it was found that many similar routines and logics were repeated under two main subroutines ‘Something scheduled’ and ‘Nothing scheduled’. It was shown that retaining the ‘Nothing scheduled’ routine with added corrections and RSC had eliminated the requirement of routine ‘Something scheduled’. This significantly simplified the algorithm. This new improved N_store algorithm was referred to as N-Store_I. The simulation results showed that the performance of N_Store_I matches with the performance of Algorithm A in the percentage of tagged cells, percentage of excess cells and percentage of police rates. This shows that a police rate algorithm that is loss less, simple to implement and equivalent to Algorithm A is possible.
7.2 Future work

The ABR policer operation is complicated, because of the variation in the round trip delay (RTD) and hence feedback delay. This is a key factor that influences the performance of a policer. In this thesis, the performance model is built on some assumptions, such as a single ABR source, choice of $\tau_2$ and $\tau_3$ based on link performance and round trip delay variation limited to LAN, MAN and WAN environments. These assumptions were made to simplify the model for the purpose of comparison of the algorithms. The absolute values of the tagged cells and excess cells may significantly vary with each scenario.

As an extension to this thesis, it is suggested that quantitative measurements be carried out with more wide ranging values of $\tau_2$ and $\tau_3$, with multiple sources and worst case scenarios such as 3 or more hops and long propagation delays. Since it is now possible to quantify the excess cells, it would be worth studying the performance of the network with such excess cells.
References


[7] Llorenç Cerda and Olga Casals, Polytechnic University of Catalonia, "Improvements and performance study of the Conformance definition for the ABR service in ATM networks" October 22, 1996


<table>
<thead>
<tr>
<th>Reference</th>
<th>Description</th>
</tr>
</thead>
</table>
References


APPENDIX 1 - RM Cell Structure

(Reproduced TM specification [1])

The details of the RM cell fields and their bit positions are given in table AP1.1. & AP1.2

<table>
<thead>
<tr>
<th>FIELD</th>
<th>OCTET #</th>
<th>BIT(s)</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>1st - 5th</td>
<td>all</td>
<td>ATM Header</td>
</tr>
<tr>
<td>ID</td>
<td>6th</td>
<td>all</td>
<td>Protocol ID</td>
</tr>
<tr>
<td>DIR</td>
<td>7th</td>
<td>8th</td>
<td>Direction</td>
</tr>
<tr>
<td>BN</td>
<td>7th</td>
<td>7th</td>
<td>BECN Cell</td>
</tr>
<tr>
<td>CI</td>
<td>7th</td>
<td>6th</td>
<td>Congestion</td>
</tr>
<tr>
<td>NI</td>
<td>7th</td>
<td>5th</td>
<td>No Increase</td>
</tr>
<tr>
<td>RA</td>
<td>7th</td>
<td>4th</td>
<td>Request/Ackn</td>
</tr>
<tr>
<td>Reserved</td>
<td>7th</td>
<td>3rd-1st</td>
<td>Reserved</td>
</tr>
<tr>
<td>ER</td>
<td>8th - 9th</td>
<td>all</td>
<td>Explicit Cell</td>
</tr>
<tr>
<td>CCR</td>
<td>10th - 11th</td>
<td>all</td>
<td>Current Cell</td>
</tr>
<tr>
<td>MCR</td>
<td>12th - 13th</td>
<td>all</td>
<td>Minimum Cell</td>
</tr>
<tr>
<td>QL</td>
<td>14th - 17th</td>
<td>all</td>
<td>Queue Length</td>
</tr>
<tr>
<td>SN</td>
<td>18th - 21st</td>
<td>all</td>
<td>Sequence</td>
</tr>
<tr>
<td>Reserved</td>
<td>22nd - 51st</td>
<td>all</td>
<td>Reserved</td>
</tr>
<tr>
<td>Reserved</td>
<td>52nd</td>
<td>16th - 11th</td>
<td>Reserved</td>
</tr>
<tr>
<td>CRC - 10</td>
<td>52nd and 53rd</td>
<td>10th-1st &amp; all</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

Table AP1.1: The RM cell structure

<table>
<thead>
<tr>
<th>Field</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATM Header</td>
<td>5 Octets</td>
</tr>
<tr>
<td>RM Protocol Identifier</td>
<td>1 Octets</td>
</tr>
<tr>
<td>Message Type  (flag bits)</td>
<td>1 Octets</td>
</tr>
<tr>
<td>ER</td>
<td>2 Octets</td>
</tr>
<tr>
<td>CCR</td>
<td>2 Octets</td>
</tr>
<tr>
<td>MCR</td>
<td>2 Octets</td>
</tr>
<tr>
<td>QL</td>
<td>4 Octets</td>
</tr>
<tr>
<td>SN</td>
<td>4 Octets</td>
</tr>
<tr>
<td>Reserved</td>
<td>30 Octets</td>
</tr>
<tr>
<td>Reserved (6 bits) + CRC 10</td>
<td>2 Octets</td>
</tr>
</tbody>
</table>

Table AP1.2: The RM cell field size

<table>
<thead>
<tr>
<th>OCTET</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>DIR</td>
</tr>
<tr>
<td>7</td>
<td>BN</td>
</tr>
<tr>
<td>6</td>
<td>CI</td>
</tr>
<tr>
<td>5</td>
<td>NI</td>
</tr>
<tr>
<td>4</td>
<td>RA</td>
</tr>
<tr>
<td>3</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

Table AP1.3: The RM cell field setting
DIR = 0 for forward RM cells and 1 for backward RM cells

BN = 1 for Non Source Generated (BECN) RM cells and 0 for Source Generated RM cells

CI = 1 to indicate congestion and 0 no congestion

NI = 1 to indicate no additive increase allowed and 0 not allowed

RA - Not used for ABR

Description of RM - cell fields:

**Header** - The first five bytes of an RM - cell are the standard ATM header with PTI= 110 for a VCC, and additionally VCI = 6 for a VPC

**ID** - The protocol ID. The TM has assigned this field to be set to 1 for the ABR service.

**Bit Flags:**

**DIR** - Forward = 0 and Backward = 1. The DIR field indicates the direction of the RM- cell with respect to the data flow which it is associated with. The source sets DIR= 0 and the destination sets DIR =1. A network element should not modify the direction bit except when turning around an RM-cell.

**BN** - A Backwards Explicit Congestion Notification (BECN) RM cell may be generated by the network or the destination. When it does so, it must set BN = 1 to indicate the cell is not source generated, and DIR= 1 to indicate the backward flow. Source generated RM - cells should be initialised with BN = 0

**CI** - The Congestion Indication (CI = 1) is used to cause the source to decrease its ACR. The source sets CI = 0 when it sends a RM - cell. Setting CI = 1 is typically how destination indicate that EFCI has been received on a previous data cell. A switch can also set CI = 1 when it is congested but
should not change the CI bit from 1 to 0. Source generated RM - cells should be initialised with CI = 0.

**NI** - The No Increase bit is used to prevent the source from increasing its ACR. It is typically used when a switch senses impending congestion. A switch should not change NI from 1 to 0. Source generated RM - cell should initialise NI with 0.

**RA** - Not used for ATM form the ABR. Source generated RM - cell should initialise RA with 0.

**ER** - The explicit Rate is used to limit the source ACR to a specific value. It is initially set by the source to requested rate (such as PCR). It may be subsequently reduced by any network element in the path, to a value that it can support.

**CCR** - The Current Cell Rate field is set by the source to a rate ≤ ACR when it generates a forward RM cell. It sends the entire in rate CLP=0 cells at this rate. The network may use CCR value to calculate ER.

**MCR** - The MCR field is set by the source to a negotiated value for the connection, which may be used by switches to facilitate bandwidth allocation. If the switch finds the value of MCR set in the RM cell is inconsistent with the signalled value, then it may correct the value to the one signalled.
Appendix 2.1 Cyclic N_Store Algorithm

**RECEPTION PROCESS**

- **CR:** Current Rate, **CT:** Current rate schedule
- **NR:** Next Rate, **NT:** Next rate schedule
- **AT:** RM cell arrival time
- **PVR:** Previous Rate, **PVT:** Previous rate schedule
- **PACR:** Feedback rate ER is stored in PACR
- **PACRM:** Previous maximum PACR
- **TSA:** Time sequence adjustment

```
Start

TSA=0 ? N
    TSA = 0

RM Cell Received?
    Y
    Wait
    Y
    Pointer Process
    N

PACR = New rate (RM cell)
AT = RM cell arrival time

CT > AT ? N
    Y
    Something Scheduled

PACR ≥ PACRM N

Rate decrease

CR = PACRM

PACR < NR Y
    NT=AT+τ₂
    N

NR=PACR

Process Ends
```

Fig. AP2.a
Appendix 2 Cyclic N Store Algorithm

ENVELOPE PROCESS – Rate increase

Rate Increase

PACRM = PACR

(\(AT + \tau_3\)) > CT

Y

NR = PACR

N

Schedule New rate first

tsaf = 0, CR = PACR, NR = PACR

PACR ≥ PVR

N Or (n>2 and PVT=0)

Y

PACR ≥ PR

Y

N

CT = AT + \(\tau_3\)

Is Envelop Pointer Idle?

N

Y

CT < PVT?

Y

N

Rate Seq.Check (RSC)

tsaf = 0 ?

N

Y

tsaf = 0

 NR = 0

Process Ends

PR: Police rate

tsaf: rate sequence check flag

rscf: time sequence check flag

n: Number of stores

If

CT = NT

Or NT = 0

Or NT > AT + \(\tau_3\)

N

Y

NT = AT + \(\tau_3\)

Process Ends

Retain old Schedule

rscf = 1 ?

N

Y

rscf = 0

CT = AT + \(\tau_2\)

Improvement

Add RSC to avoid lower rates policing higher rates

Fig.AP2.b
ENVELOPE PROCESS - Nothing Scheduled

Correction 1
CR should be stored with
CR=PACR irrespective of
whether NT=0 or NT>0. Hence
decision box should be removed

Nothing Scheduled

NR=PACR

NT <> 0

Y

CR=PACR
PACRM=PACR

N

PACR>PVR
Or (n>=2 and PVT=0)

N

Delay the Rate

CT=AT+τ₂

Y

Don't Delay the Rate

PACR ≥ PR

Y

CT=AT+τ₃

N

Pointer Idle?

N

Pointer terminated

N

rs cf=1 ?

Y

rs cf=0

CT=AT+τ₂

N

CT<PVT

Y

TSA

Rate Seq. Check (RSC)

Process Ends

Fig. AP2.c

PR: Police rate
PT: Police rate schedule
RSC: Rate sequence check
rs cf: rate sequence check flag
ts af: time sequence check flag

Improvement
Add RSC to avoid
lower rates policing
the higher rates
TIME SEQUENCE ADJUSTMENT - TSA

TSA - Time Sequence Adjustment

Wake up if pointer is idle

TSA = 1

Move pointer backward

\[
\begin{align*}
CR &= NR \\
PACRM &= PVR \\
P_n &= P_c, P_c &= P_p \\
Ps &= P_n + 1 \\
P_d &= P_d - 1
\end{align*}
\]

Cyclic check for Ps

\[
\begin{align*}
N & \quad Ps > n \\
Y &
\end{align*}
\]

Ps = 1

Cyclic check for Pp

\[
\begin{align*}
N & \quad Pp < 1 \\
Y &
\end{align*}
\]

Pp = n

Correction 2
Unschedule next store. Add NT = 0 in the routine.

NT = 0

Process Ends

Fig. AP2.d
DELIVERY PROCESS

Start
Delivery

Initialise
Counter
del_s = 1

PT(del_s)=0? Y
N

PT(del_s) - Realtime Y
> τ_3/n-1?
N

PR =PR(del_s)
PR(del_s) = 0

Move pointers
forward

del_s = del_s + 1

N del_s > n
Y

del_s = 1

Process Ends

Wait until new police rate schedule arrives

Wait for τ_3/n-1

PR: Police rate
PT: Police rate schedule
del_s: Delivery pointer

Fig. AP2.e
Appendix 2  Cyclic_N_Store Algorithm

POINTER PROCESS

1. Pointer Process
2. Wait for \( \tau_3/n-1 \)
3. If ST > 0?
   - If Y: Wait until a vacant store is made available by delivery process
   - If N: CT > 0
     - If Y: Move pointers forward
       - For n > 2
         - \( P_p = P_c \)
         - \( P_c = P_n \)
         - \( P_n = P_s \)
         - \( P_s = P_n + 1 \)
         - NR = CR
         - PACRM = CR
     - If N: \( P_s > n \)
       - If Y: \( P_s = 1 \)
3. Process Ends

Fig. AP2.f
RATE SEQUENCE CHECK (RSC)

Rate Sequence Check
RSC

N
CR > PVR
Y

PVT ≥ AT + τ3
N

Y
rscf=1
TSA

N
TSA=0

Return

Fig. AP2.g
Appendix 2.2 - Pseudo code - Cyclic N_Store Algorithm

(Majority of the contents are reproduced from original article [2])

Initialisation:

\[
\text{time\_seq\_adj} := \text{false}; \\
iacr := 2.0; \quad /* \text{Any rate to start it off}\*/ \\
pacr\_max := \text{iacr}; \\
\text{store(current).initialise(iacr)}; \\
\text{store(next).initialise(iacr)}; \\
\text{store(previous).initialise(iacr)}; \\
\text{store(current).set\_time(10000, envelope\_id)}; \\
\text{store(next).set\_time(0.0, envelope\_id)}; \\
\]

The Envelop pointer process

BEGIN
PROCEDURE init_pointers;
BEGIN
previous := 0;
current := 1;
next := 2;
s := circular\_add(next, n);
END;

\textbf{Increment Pointer:} /*After the reception of RM cell the Pointer process is activated to move the pointer to next store*/

integer s, current, next, previous;
WHILE TRUE DO /* Await activ by recep, when 1st rm cell arrives*/
BEGIN
HOLD(\tau_3/(n-1)); /* Hold for this time period to provide a minimum gap between successive RM cells*/
WHILE store(s).timenow > 0 DO /*there is already a value in the store, so sleep until activated by DELIVERY when it has delivered this value*/
BEGIN
IF (store(current).timenow) > 0 THEN /*At least one b/w RM cell has arrived*/
move\_pointers\_forward;
END--WHILE;

PROCEDURE move\_pointers\_forward; /*This proc. moves the pointers forward, and copies the last rate received into store(next).*
Appendix 2  Cyclic_N_Store  Algorithm

BEGIN
IF n > 2 THEN /* N = 2 is a special case;*/
    previous := current;
current := next;
next := s;
s := circular_add(next,n);
store(next).set_pacr(store(current).pacr);
pacr_max := store(current).pacr;
END--move_pointers_forward;

**Envelop Process:** Starts after the arrival of RM cell with new feedback rate and its arrival time

WHILE TRUE DO /*either wait for a message with new rate, or perform algorithm a second time using current message pacr and arrival_time*/
BEGIN
IF NOT time_seq_adj THEN /* If time_seq_adj flag is false then wait for new message with new rate */
    wait_for_new_message
ELSE /* Otherwise clear the flag and repeat the envelope process */
    time_seq_adj := false;
INSPECT message DO /*Check arrival_time captured during receiving process with the current time. This time will be the*/
IF store(current).time > arrival_time THEN /* from the previous message if the process is repeated for TSA iteration */
    something_scheduled /*something is scheduled in current store*/
ELSE
    nothing_scheduled;
END--inspect;
END--while;
END--ENVELOPE_pc;

**PROCEDURE something_scheduled:**
begin
IF pacr >= pacr_max THEN /*its an increase*/
    rate_increase
ELSE
    rate_decrease;
end--something_scheduled;

**PROCEDURE rate_increase:**
begin
    pacr_max := pacr; /* new value for pacr_max */
    IF (arrival_time + τ₃) > store(current).time THEN
        something_is_scheduled_earlier
    ELSE
        schedule_newrate_first;
end--rate_increase;

END--move_pointers_forward;
PROCEDURE something_is_scheduled_earlier;
begin
store(next).set_pacr(pacr); /* If Current store is already scheduled earlier
than the new schedule */
IF (store(current).time = store(next).time) OR /* Then PACR will be stored in next store. */
(store.next).time = 0) OR
(store(next).time > arrival_time + T) THEN /* If next store is unscheduled or
scheduled same as current schedule or
scheduled */
store(next).set_time(arrival_time + T); /* later than arrival time + T then it is scheduled
to arr.time + T */
end—something_is_scheduled_earlier; /*If it is scheduled earlier than arr.time + T then
old schedule is retained */

PROCEDURE schedule_newrate_first;
Begin
boolean tsa_flag;

.tsaf=-false; /* reset flag for time seq. Adjustment */
store(current).set_pacr(pacr); /* newrate is scheduled first*/
store(next).set_pacr(pacr); /*store(next) must always hold last reed, rate*/
IF (pacr >= store(previous).pacr) OR (n > 2 AND store(previous).time = 0)
THEN
Begin
*now check against the policer rate*/
IF pacr > store(0).pacr THEN /*its an increase, don’t delay it*/
store(current).set_time(arrival_time + T)
ELSE IF NOT envelope_pointer.idle AND NOT (envelope_pointer.terminated) THEN
{ IF rscf = 1 THEN /* Don’t schedule for Rate seq. check (RSC)*/
srscf=0
store(current).set_time(arrival_time + T); /*envelope pointer is not asleep,
it’s a decrease hold on for T*/
}
IF store(current).time < store(previous).time THEN
Begin
	tsa_flag = true;

*Set to avoid doing next assignment. The flag
indicates , it has gone through Time seq. adjus */
/*collapse the stores, won’t happen for n=2*/
End If store (current)....
ELSE rate_sequence_check;
End If PACR >=.......
If not tsa_flag THEN /* If tsaf flag is reset then only clear next store
time */
store(next).clear_time;
ELSE tsa_flag = false;
/* If tsaf flag is set then just reset the tsaf flag */
End—schedule_newrate_first;

PROCEDURE rate_sequence_check;
Begin
* Check RSC only if current rate is greater */
IF (store(current) > store(previous)) THEN /* than the previous rate and previous schedule */
Begin
/* is greater than arrival time + T */
Appendix 2 Cyclic_N_Store Algorithm

IF (store(previous).set_time ≥ (arrival_time + τ₃)) THEN

Begin

- rscf = 1;
- time_sequence_adjustment;
- TSA = 0;
End;

End — rate_sequence_check;

PROCEDURE time_sequence_adjustment:

Begin

IF (envelope_pointer.idle) AND NOT (envelope_pointer.terminated) THEN

Reactivate the envelope_pointer AFTER current

_ time_seq_adj = true;

_envelope_pointer.move_pointers_backward; /* Set the time_sequence_adj flag in order to

store(next).clear_time; /* Unschedule the next store */

End — time_sequence_adjustment;

PROCEDURE move_pointers_backwards:

BEGIN

store(current).set_pacr(store(next).pacr);

pacr_max := store(previous).pacr;

next := current;

previous := previous;

next := circular_add(next, n);

previous := circular_dec(previous, n);

END — move_pointers_backwards;

PROCEDURE rate_decrease:

begin

store(current).set_pacr(pacr_max);

IF pacr < store(next).pacr THEN

store(next).set_time(arrival_time + X τ₂);

store(next).set_pacr(pacr);

end — rate_decrease;

PROCEDURE nothing_scheduled:

begin

store(next).set_pacr(pacr);

if store(next).time ≤ 0 THEN

Begin

store(current).set_pacr(pacr);

pacr_max := pacr;

end;

IF (pacr > store(previous).pacr) OR (n > 2 AND store(previous).time = 0) THEN

dont_delay_rate  

*If so, schedule the rate immediately*/
Appendix 2  Cyclic_N_Store Algorithm

ELSE
    delay_rate;
end—nothing_scheduled;

PROCEDURE dont_delay_rate;
begin
    /*if rate is > than policer rate schedule it for time + \( \tau_3 \) else time + \( \tau_2 \)*/
    IF pacr \( \geq \) store(0).pacr THEN
      store(current).set_time(arrival_time + \( \tau_3 \))
      /* must check this rate against the policer rate*/
    ELSE IF NOT envelope_pointer.idle AND NOT (envelope_pointer.terminated) THEN
      IF  \( rscf = 1 \) THEN
        /* Don't schedule for Rate seq. check (RSC) */
        rscf=0
      END;
      store(current).set_time(arrival_time + \( \tau_2 \));
      /*if envelope pointer is asleep then there are no more vacant stores in which to build up a rate*/
      IF store(current).time < store(previous).time THEN
        /*stores may need to be 'collapsed' now*/
        time_sequence_adjustments;
    END;
end—dont_delay_rate;

PROCEDURE delay_rate;
begin
    /*no need to check policer rate, just set delivery time to arr_time + \( \tau_2 \)*/
    store(current).set_time(arrival_time + \( \tau_2 \));
end—delay_rate;

THE DELIVERY PROCESS
BEGIN
    del_s:= 1;
    WHILE TRUE DO
      BEGIN
        while store(del_s).timenow_is_zero do
          go to sleep until reactivated;
        WHILE (time <= simperiod + .01) AND THEN
          /* Loop until the real time crosses the delivery schedule */
          (ABS (store(del_s).timenow - time ) > .00001) AND
          (store(del_s).timenow > time) DO
            begin
              if (store(del_s).timenow - time > (\( \tau_3/(n-1) \))) then
                hold(\( \tau_3/(n-1) \))
                /* hold for standard time period ;*/
              else
                hold(store(del_s).timenow -time);
                /* hold until current time = store time ;*/
              end—while(ABS);
            set policer rate to store 0 rate;
            clear store 0 time;
            wake up envelope_pointer if necessary;
            if envelope_pointer.current_equal(del_s) then
              envelope_pointer.move_pointers_forward;
              del_s:= circular_add(del_s,n);
            END—while;
            END—DELIVERY_P;
    END—of Algorithm
Appendix 3.1 - Flow Chart - Cyclic N_Store_I Algorithm

RECEPTION PROCESS

Fig. AP3.a
Cyclic N Store I Algorithm

ENVELOPE PROCESS

Nothing Scheduled

NR=PACR
CR=PACR

Delay the Rate

PACR>PVR
Or (n>2 and PVT=0)

Don’t Delay the Rate

PACR ≥ PR

Pointer Idle?

Pointer terminated

CT < PVT

Rate Seq. Check (RSC)

Process Ends

CT=AT+ τ2

CT=AT+ τ3

CT=AT+ τ2

rscf=0

rscf=1?

TSA

Fig. AP3.b
Appendix 3  Cyclic_N_Store_I Algorithm

N_Store_I Algorithm
TIME SEQUENCE ADJUSTMENT - TSA

TSA - Time Sequence Adjustment

Wake up if pointer is idle

TSA = 1

Move pointer backward

CR=NR
PACRM = PVR
Pn=Pc
Pc = Pp
Ps = Pn + 1
Pp = Pp - 1

Cyclic check For Ps

N  Ps > n
Y

Ps = 1

Cyclic check for Pp

N  Pp < 1
Y

Pp = n

NT=0

Process Ends.
Cyclic N_Store_I Algorithm

DELIVERY PROCESS

Start Delivery

Initialise Counter
del_s = 1

PT(del_s) = 0? Y

N

PT(del_s) - Realtime > τ_s/n-1? Y

Wait for τ_s/n-1

N

PR = PR(del_s)
PR(del_s) = 0

Move pointers forward

del_s = del_s + 1

N del_s > n Y

del_s = 1

Process Ends
Appendix 3 Cyclic N Store I Algorithm

POINTER PROCESS

- Pointer Process
- Wait for $\tau_y/n - 1$
- ST > 0?
  - N
  - CT > 0
    - N
    - Move pointers forward
      - For $n > 2$
        - Pp = Pc
        - Pc = Pn
        - Pn = Ps
        - Ps = Pn + 1
        - NR = CR
      - Y
      - Ps > n
        - N
        - Ps = 1
  - Y
- Wait until a vacant store is made available by delivery process

Process Ends

Fig. AP3.e
Cyclic N_Store_I Algorithm

**RSC = RATE SEQUENCE CHECK**

- Rate Sequence Check (RSC)
  - If \( CR > PVR \) then Yes
  - If \( PVT \geq AT + \tau_3 \) then \( rscf=1 \) and TSA
  - Return
Appendix 3.2 Pseudo Code - Cyclic N_Store_I Algorithm

(Majority of the contents are reproduced from original article [2])

Initialisation:

time_seq_adj := false;
iacr:= 2.0 ; /* Any rate to start it off*/
pacr_max:=iacr;
    store(current).initialise(iacr);
    store(next).initialise(iacr);
    store(previous).initialise(iacr);
    store(current).set_time(10000, envelope_id);
    store(next).set_time(0.0, envelope_id);

The Envelop pointer process

BEGIN
PROCEDURE init_pointers;
BEGIN
    previous := 0;
    current := 1;
    next := 2;
    s:= circular_add(next,n);
END;

Increment Pointer:
/* After the reception of RM cell the Pointer process is activated to move the pointer to next store*/

integer s, current, next, previous;
WHILE TRUE DO /* Await activ by recep, when 1st rm cell arrives*/
BEGIN
    HOLD(T3/ (n-1)); /* Hold for this time period to provide a minimum gap between successive RM cells*/
    WHILE store(s).timenow > 0 DO /*there is already a value in the store, so sleep until activated by DELIVERY when it has delivered this value*/
        IF (store(current).timenow) > 0 THEN move_pointers_forward; /*At least one b/w RM cell has arrived*/
    END--WHILE;

PROCEDURE move_pointers_forward; /*This proc. moves the pointers forward, and copies the last rate received into store(next).*/
BEGIN
    IF n > 2 THEN /* N = 2 is a special case;*/
Appendix 3 Cyclic_N_Store_I Algorithm

previous := current;
current := next;
next := s;
s:= circular_add(next,n);
store(next).set_pacr(store(current).pacr);
END--move_pointers_forward;

**Envelop Process:** Starts after the arrival of RM cell with new feedback rate and its arrival time and the pointers are moved forward

WHILE TRUE DO
BEGIN
   IF NOT time_seq_adj THEN
      wait_for_new_message
   ELSE
      time_seq_adj := false;
   INSPECT message DO
      nothing_scheduled;
   END--.inspect;
END--while;
END--ENVELOPE_pc;

PROCEDURE rate_sequence_check;
Begin /* Check RSC only if current rate is greater */
   IF (store(current) > store(previous)) THEN /* than the previous rate and previous schedule */
      Begin /* is greater than arrival time + τ3 */
         IF (store(previous).set_time ≥ (arrival_time + τ3)) THEN
            Begin
               rscl = 1;
               time_sequence_adjustment;
            End;
         End;
      End;
   End—rate_sequence_check;

PROCEDURE time_sequence_adjustment;
Begin
   IF (envelope_pointer.idle) AND NOT (envelope_pointer.terminated) THEN
      Reactivate the envelope_pointer AFTER current
      time_seq_adj = true;
      envelope_pointer.move_pointers_backward;
      IF envelope_pointer.asleep THEN /* Set the time_sequence_adj flag in order to */
         /* repeat the envelope process without */
         /* taking the new message. */
         /* This is repeated as long as the time */
awaken it; sequence adjustment needed */
end--time_sequence_adjustment; /*bring env_pointer back to life*/

PROCEDURE move_pointers_backwards:
BEGIN /* This proc is never called when n = 2;*/
  store(current).set_pacr(store(next).pacr);
  pacr_max := store(previous).pacr;
  next := current;
  current := previous;
  s := circular_add(next,n);
  previous := circular_dec(previous,n);
END--move_pointers_backwards;

PROCEDURE nothing_scheduled:
begin /*set rates in current, next, and pacr_max*/
  store(next).set_pacr(pacr); /*check whether new rate is greater than the rate
  store(current).set_pacr(pacr); held in store previous, n = 2 is special case*/
  IF (pacr > store(previous).pacr) OR (n > 2 AND store(previous).time = 0) THEN
    dont_delay_rate /*If so schedule the rate immediately*/
  ELSE
    delay_rate;
end--nothing_scheduled;

PROCEDURE dont_delay_rate:
begin /*if rate is > than policer rate schedule it for time + \tau_3 else time + \tau_2*/
  IF pacr := store(0).pacr THEN
    /* must check this rate against the policer rate*/
    store(current).set_time(arrival_time + \tau_3)
  END; /*Don't schedule for Rate seq. check ( RSC)*/
  ELSE IF NOT envelope_pointer.idle AND NOT (envelope_pointer.terminated) THEN
    store(current).set_time(arrival_time + \tau_2); /*if envelope pointer is asleep then
    END; there are no more vacant stores
    IF store(current).time < store(previous).time THEN /*stores may need to be 'collapsed' now*/
      time_sequence_adjustment;
    ELSE rate_sequence_check;
  END--dont_delay_rate;

PROCEDURE delay_rate:
Begin
  store(current).set_time(arrival_time + \tau_2); /*no need to check policer rate, just set
delivery time to arr_time + \tau_2*/
  end--delay_rate;

THE DELIVERY PROCESS
BEGIN
  del_s := 1;
  WHILE TRUE DO
BEGIN
    while store(del_s).timenow_is_zero do /* If the schedule is zero, then no police rate
    END--delivery_rate; available for delivery */
go to sleep until reactivated;  
WHILE (time <= simperiod + .01) AND THEN /* Loop until the real time crosses the delivery schedule */
(ABS (store(del_s).timenow - time) > .00001) AND
(store(del_s).timenow > time) DO
begin
  if (store(del_s).timenow - time) > (τ₃/(n-1)) then
    hold(τ₃/(n-1)) /* hold for standard timeperiod;*/
    else
      hold(store(del_s).timenow - time) ; /* hold until current time = store time;*/
    end--while(ABS);
  set policer rate to store 0 rate; /* Copy the the policer rate to store(0 )*/
  clear store 0 time;
  wake up envelope_pointer process if necessary; /* step to next store with valid entry, moving pointers if nec.*/
  if envelope_pointer.current_equal(del_s) then
    envelope_pointer.move_pointers_forward;
    del_s:= circular_add(del_s,n);
  END--while;
  END--DELI Very;
END--N_Store_I Algorithm
APPENDIX 4 - RTT Estimation - 2Pass method

(Reproduced the portion of article [6])

The estimation of Round Trip Time (RTT) is not specified in TM 4.0. The RTT may be estimated as the sum of propagation delays, transmission delays and queuing (cell scheduling) delays in one round trip of RM cell. This is estimated through signalling during establishment of the call. The general recommended method is a 2 pass method [6] which is described below. Refer to fig. AP5.1

In the first pass, as the call setup starts out, the source sets up four separate call negotiation parameter groups. It sets a trial RTT for each group selecting four values, which span the range (e.g. 4 ms, 16 ms, 64 ms, & 256 ms). As the call proceeds, the switches negotiate the parameters for both the forward and the backward paths using the proposed RTT as the total round trip from (virtual) source to (virtual) destination. All switches compute as if they were Non-BECN switches and use the total RTT. As the destination (or virtual destination) is reached, the real RTT is finally known (twice the accumulated delay). The destination then selects the best remaining parameters for the group whose trial RTT is closest to, but less than, the real RTT. For example if the real RTT is 40ms then the destination selects the parameter 64ms. If this is a virtual destination, the parameters should be stored at the destination until the second pass returns to this point. A new set of trial RTT groups should then be started by the virtual source, and the call proceeds.

In the second pass when the real destination is reached, and the call is turned around, the final segment parameters should be loaded into the connect message. When the connect message reaches a virtual source, the parameters are delivered to the source. The process then repeats by the next segments parameters being loaded into the connect message and the call proceeding
Appendix 5 - RTT Estimation – 2 Pass Method

until the real source is found. The last segment parameters are given to the real source and the call setup is complete. For example, if the real RTT turns out to be 32 ms, and the trial values included 16 ms and 64 ms, then with the two pass system the closest higher RTT would be 64 ms

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Procedure for Non-BECN Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>Collect RTT. Save at each Virtual Dest.</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>Compute Parameters for Forw. &amp; Backwd paths for 4 trial RTT’s. (4,16,64,256 ms)</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>At destination, select trial closest to RTT.</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>Deliver Forward &amp; Backward Parameters.</td>
</tr>
</tbody>
</table>

Fig AP4.1: RTT estimation process

-------------------End of the Thesis-------------------