

# Traffic Dispersion and Its Impact on ATM Protocol Functions

Eva Gustafsson\*  
Ericsson Radio Systems AB  
S-164 80 Stockholm, SWEDEN  
Eva.Gustafsson@era-t.ericsson.se

Gunnar Karlsson  
Swedish Institute of Computer Science  
Box 1263, S-164 28 Kista, SWEDEN  
gk@sics.se

## Abstract

*Traffic dispersion, which means spreading traffic from a source over multiple disjoint paths through a network, has gained interest for the handling of bursty multimedia traffic. The strategy has been shown to improve network performance and to increase security and tolerance to faults. Our study focuses on the implementation of the technique in an ATM network, and discusses the protocol functions and signalling procedures necessary to support traffic dispersion. These include connection establishment and release for multiple paths, and the dispersion of cells at the sender with resequencing at the receiver. There are also simulation results included to show how dispersion affects the delay a cell suffers through a network. Lastly, we give an example of pseudocode for implementing resequencing and decoding of dispersed cells in a receiving AAL.*

## 1. Introduction

Traffic dispersion denotes a technique of spreading traffic over multiple disjoint paths from source to destination, transmitting it in parallel through the network. The technique has gained attention during the last years as an efficient means of handling bursty and strongly correlated traffic from data and multimedia applications [6]. By splitting a traffic stream, the correlation in it can be reduced and accordingly the queuing delay and loss performance in network switches can improve. Dispersion also provides efficient use of forward erasure correction, since losses due to link faults on different paths are independent of each other. Transmitting redundant information on one or several of the paths can thus improve the network fault tolerance. This, together with the difficulty to eavesdrop on a multipath transmission, makes traffic dispersion efficient in improving network security and reliability.

In particular, traffic dispersion has the potential to improve the multiplexing performance in ATM networks. The Asynchronous Transfer Mode, ATM, which is the recommended transfer mechanism for B-ISDN, establishes virtual connections over which data is transmitted in fixed-size packets called *cells*. Each cell carries 48 octets of payload and a 5-octet header, which contains routing and control information for the cell. The traffic from several virtual connections is statistically multiplexed on the network links to maximize the capacity utilization. This may result in occasional cell losses due to buffer overflow in the switches, particularly when the traffic is bursty and strongly correlated over long time periods. Data and multimedia traffic, which is anticipated to occupy much of the resources in B-ISDN, has been shown to exhibit such behaviour

[16][17]. Consequently, the capability of traffic dispersion to alleviate the effects of large traffic bursts makes it a strong competitor to ordinary traffic shaping. Although several studies have shown the positive effects of traffic dispersion [6], there has been no thorough discussion on how to implement it in an ATM network, nor on how it affects the protocol functions. The aim of this study is to focus on these issues.

The B-ISDN protocol reference model defines the different layers of ATM: the *physical layer*, the *ATM layer* and the *ATM adaptation layer (AAL)* [4]. The physical layer performs physical medium dependent functions [13], while the ATM layer is responsible for the multiplexing and demultiplexing of cells [9]. At the sending side the ATM layer adds a header to each cell information field received from the AAL, and at the receiving side the cell headers are removed. The AAL is divided into a *segmentation and reassembly (SAR)* sublayer and a *convergence sublayer (CS)*, and its basic function is to adapt the services provided by the ATM layer to the requirements of the higher layer [10]. At the sending side, the SAR sublayer cuts the information stream arriving from the higher layer into pieces, suitable for the information field of an ATM cell. These information fields are at the receiving side reassembled to form the original information flow of the higher layer. In order to minimize the number of AAL protocols, a certain service classification has been proposed, and different AAL types (AAL type 1 to 5), reflecting the service classification, have been defined [11].

Since the functions of the physical layer only deal with the physical transmission of cells, they do not need to be affected when traffic dispersion is implemented. Similarly, the ATM layer should remain unaffected. The ATM layer does not make routing decisions; it only inserts existing routing information in the cell headers. When delivered to the ATM layer, the cells should therefore already be dispersed. Consequently, the AAL should be the layer responsible for the functions supporting traffic dispersion. When it comes to practical implementation of ATM, the AAL mostly used is AAL type 5 [20]. In this study, we discuss modifications due to dispersion when applied to AAL type 5. This results in a modified AAL type 5, which we refer to as AAL type 5'.

We define the *dispersion factor*  $N$  as the number of paths used for a transmission. The traffic from a source is dispersed over  $k$  paths, where  $k \leq N$ , while the remaining  $N-k$  paths carry redundant information. In the remainder of this study, we infer dispersion to mean *cyclic dispersion* of ATM cells, since this dispersion strategy has been shown to give the best performance [7].

\*This work was performed while the author was at KTH Department of Teleinformatics; evag@it.kth.se.

In Section 2, we discuss connection establishment, and Section 3 defines protocol functions for dispersion of cells. The resequencing procedure is presented in Section 4, and coding and decoding is discussed in Section 5. Section 6 presents a pseudocode example for the receiving protocols, and Section 7 ends the study with a few concluding remarks.

## 2. ATM connection establishment

An ATM virtual connection is defined by its VPI/VCI, that is, the *virtual path identifier* and the *virtual channel identifier*, where the VCI only has local significance within a virtual path [8]. When a user wants to transmit it asks the network to establish a virtual connection. Based on the traffic descriptors declared by the user, the quality of service (QoS) required for the call and the current network load, the network decides whether to accept or deny the request [12]. When a request is accepted, the network indicates the VPCI (*virtual path connection identifier*) and/or VCI values to be used, and the virtual connection is established.

When implementing traffic dispersion, we propose that the decision of whether to use dispersion or not for each call should be in the hands of the network, and not the individual users. The network has better knowledge than an individual user of the network load and topology, and of the number of users currently employing dispersion. It can decide whether dispersion is useful or not, given the current network utilization, and it can find the number of paths which are actually available for dispersion at the time of the connection request. If the user wants to disperse the traffic for other reasons than capacity gain, for example increased security or reliability, it should explicitly ask for a dispersed transmission. In that case, the network should meet the user requirements, without consideration of whether capacity gains can be obtained or not.

Thus, the connection establishment proceeds as follows. The user sends a call request to the network. Depending on the number of available disjoint paths in the network and the demands from the user, the network decides the dispersion factor for the call. It indicates the VPCI/VCI values in the order that they should be used for the dispersion of cells, and the connection is established. In this study, we consider dispersion for point-to-point transmission, also referred to as unicast.

### 2.1. Connection establishment with dispersion

When a user requests a connection, it sends a SETUP message over an established signalling channel to the network, and the network sends a SETUP message to the called user. As part of the message, there is a *call reference*, which is used to identify the call at the local user-network interface. The call reference value is set at the beginning of a call and remains unchanged for the duration of the call. When using traffic dispersion, we let the call reference value be the same for all the connections established for a call, in order to minimize the required number of signalling messages.

Fig. 1 shows an example of the exchange of signalling messages at connection establishment [21]. All the messages contain the call reference. Each of these messages,

except the CONNECT ACKNOWLEDGE message, may also contain a *connection identifier* information element, which is used to identify the VPCI/VCI values for the connection. The sender can use this element to declare a preferred virtual connection identifier to the network, and the network uses the connection identifier information element to declare a virtual connection, defined by a VPCI and a VCI value, to the users [14].

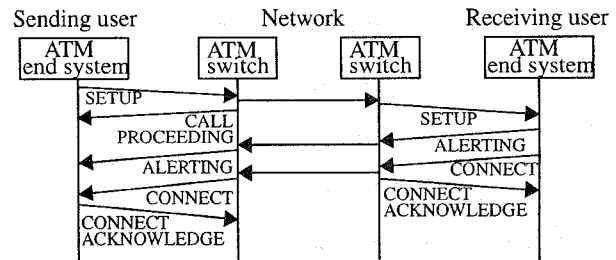


Fig. 1. Example of exchange of signalling messages at connection establishment.

In order to set up and maintain multiple connections for traffic dispersion, the connection identifier element will be extended to contain the VPCI and VCI values for all the connections used for the dispersion (Fig. 2). Connection 1 is thus defined by the VPCI/VCI values in octets 6 to 9 of the element, while connection  $N$  is defined by the VPCI/VCI values in octets  $(6 \text{ to } 9) + 4(N - 1)$ . The overhead in a signalling message due to the extended connection identifier information element for traffic dispersion is  $4(N - 1)$  octets. For example, with  $N = 5$  and a signalling message size varying from 17 to 178 octets or more without dispersion [14], the overhead ranges from below 8% up to 45%.

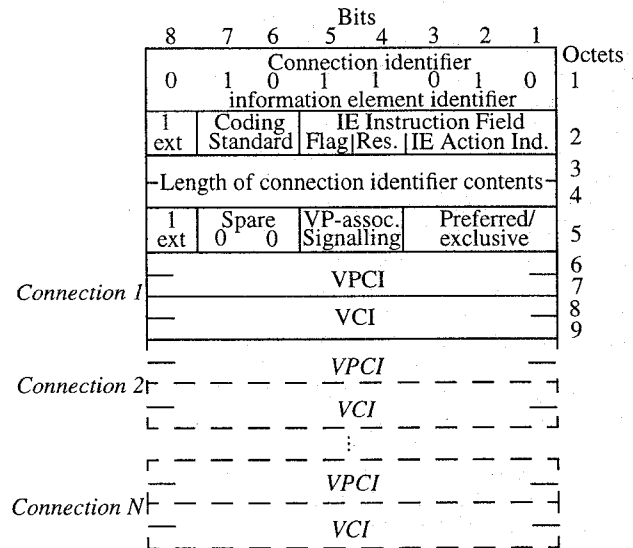


Fig. 2. Connection identifier information element used in the control messages at connection establishment with traffic dispersion.

The SETUP message also contains an *ATM traffic descriptor*, which specifies the ATM peak cell rates in the forward and backward directions. This traffic descriptor is not affected by dispersion: once the dispersion factor and the redundancy level are decided, the declared peak rate of the

source together with these values forms the base for capacity allocation on the paths.

Dispersion requirements from the user could be declared in the AAL parameters field in the SETUP message, as suggested in Fig. 3. The dispersion factor  $N$  and the type of coding is stated, so that the network can notify the receiver of what code and  $k$  value to use. If the desired number of paths is not available in the network, the network denies the request. The denial is made through the sending of a RELEASE COMPLETE message [14], with the *cause* information element indicating that the requested number of paths was not available.

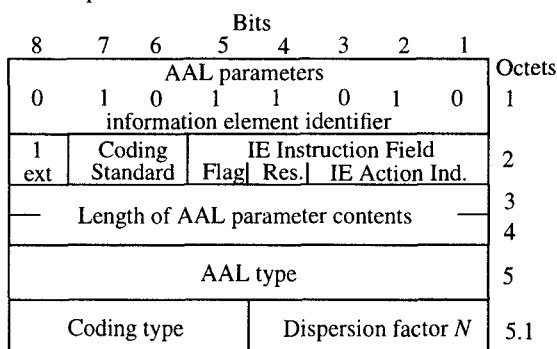


Fig. 3. AAL parameters information element (first octets).

The release of connections belonging to a call is initiated by a RELEASE message, which requests the end-to-end connections identified by the call reference to be cleared. Dispersion does not affect the signalling messages for the release procedure.

### 3. Dispersion of cells

The SAR sublayer is responsible for dispersing the SAR protocol data units over the appropriate virtual connections, as well as resequencing these units at the receiver. There is normally one SAR process for each end of a virtual connection. In order to perform dispersion though, there has to be a single SAR process for all the  $N$  virtual connections used for the transmission. The scenario is shown in Fig. 4.

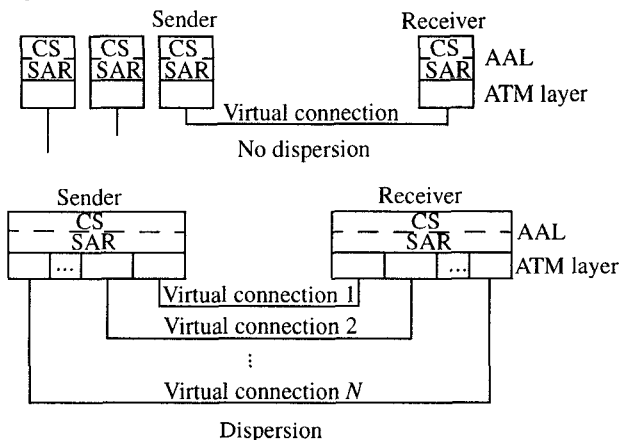


Fig. 4. How the SAR and ATM layer processes should be organized to handle multiple parallel virtual connections.

When dispersing the units cyclically over the virtual connections, the SAR process at the sender steps in round-robin order from ATM layer process to ATM layer process, delivering one SAR protocol data unit at each step. The  $N$  ATM layer processes with different VPI/VCI values are organized in the order specified at connection establishment. At the receiving side, the SAR process reads one SAR protocol data unit at a time from each ATM layer process in the same order, again defined at connection establishment. The interface between the ATM layer and the AAL at the receiver is assumed to be resequencing buffers, where units are stored until the SAR process takes them out. This means that the resequencing of the units is performed just before the reassembly at the receiving SAR sublayer. A method to implement the resequencing is described in the next section.

### 4. Resequencing

Since the connections used for a dispersed transmission may extend over paths of various lengths, cells will experience different amounts of queuing and propagation delay during their individual transfers. It may hence be difficult to determine whether an absent cell is late or lost. There could be a time limit after which an absent cell is treated as lost. However, the distance between cells might decrease in the network due to multiplexing, causing delay variations, and a later cell could be mistaken for an earlier sent cell. This would violate the ATM guarantee of orderly delivery, and it is therefore more secure to use sequence numbers.

We suggest sequence numbers to be included in AAL type 5' (Fig. 5). The resequencing of the cells is performed before the reassembly at the receiving SAR sublayer, and accordingly, the generation of sequence numbers at the sending side is performed at the SAR sublayer. As part of the resequencing procedure, the SAR sublayer also discards detected misinserted cells.

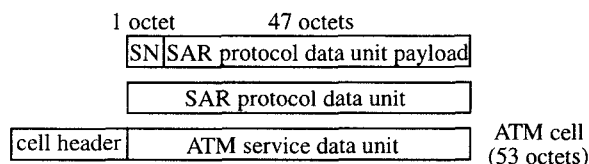
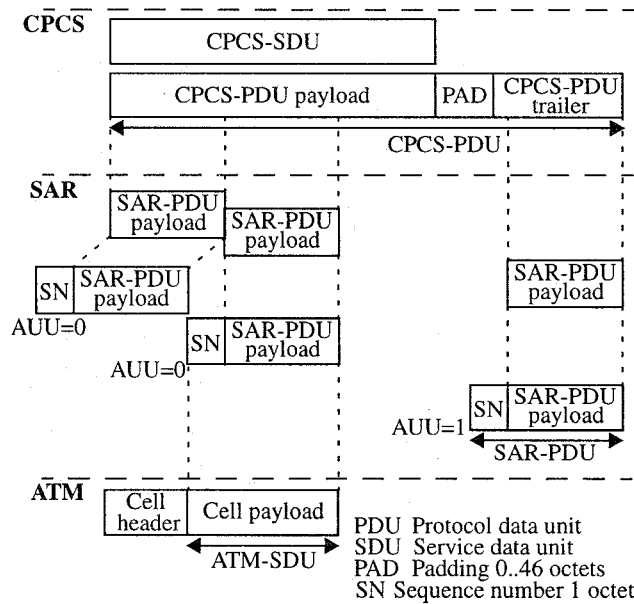


Fig. 5. The cell format for AAL type 5', including cell sequence number (SN) for resequencing of dispersed cells at the receiver.

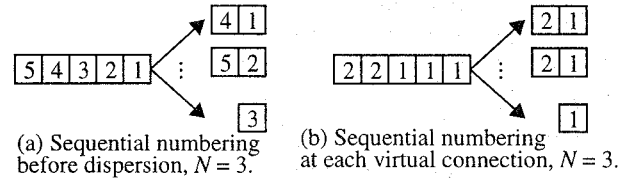
With non-dispersed transmissions over AAL type 5, the unit exchanged between the SAR sublayer and the CPCS (common part convergence sublayer) is an integral multiple of 48 octets [11]. To achieve this, a padding function provides 48 octet alignment of the CPCS protocol data unit trailer. When employing dispersion with AAL type 5', the aim of the alignment should be to achieve a CPCS protocol data unit size which is a multiple of 47 octets. This is because of the one octet sequence number added to each SAR protocol data unit. The sequence number may be chosen to be smaller, but in this study, we suggest a one octet sequence number due to the increased hardware complexity that would be the result of splitting the octet-based structure of AAL type 5. Furthermore, losses caused by link and nodal failures could be in the range of ten to hundred milliseconds before the path is restored, by protection-switch-

ing [19]. Such long outages require a substantial range of the sequence numbers per virtual connection. The sequence number employed here is able to detect the loss of up to  $2^8 - 1 = 255$  consecutive cells on a path. It introduces an overhead of 1 octet or 2.1% per cell. In return, its capability to detect lost cells may prevent retransmissions in the case of redundant dispersion. The 47 octet alignment at the CPCS and the segmentation at the SAR sublayer are illustrated in Fig. 6. The SAR sublayer utilizes the AUU (ATM-layer-user to ATM-layer-user) parameter of the ATM layer primitives to indicate that a SAR protocol data unit contains the end of a SAR service data unit.  $AUU = 1$  indicates the end of a SAR service data unit, while  $AUU = 0$  indicates the beginning or continuation of it [11]. The AUU parameter is transported transparently by the ATM layer in the payload type field of the cell header.



**Fig. 6. Exchange of units between the CPCS and the SAR sublayer for AAL type 5'. The padding provides for the CPCS protocol data unit to be an integral multiple of 47 octets. At the SAR sublayer, sequence numbers are added to the SAR protocol data units. The ATM cell payload consists of the SAR protocol data unit payload (47 octets) and the sequence number (1 octet).**

Two ways to use the sequence numbers are illustrated in Fig. 7. When numbering the protocol data units according to Fig. 7 (a), the SAR process at the receiver can verify the order of the virtual connections from the sequence numbers. If numbering according to Fig. 7 (b) is employed, there need to be separate resequencing buffers for the arriving virtual connections in order not to confuse the units. We suggest to use sequence numbering according to Fig. 7 (b), where the SAR sublayer generates a sequence number and inserts it in the next  $N$  available sequence count fields. Furthermore, we choose to employ physically or logically separate FIFO resequencing buffers at the arriving connections.



**Fig. 7. Example of how to employ sequence numbering when dispersing cells.**

#### 4.1. Resequencing delay

In the following, we discuss how dispersion affects the delay that cells suffer during transmission. The total transfer delay through a network is the sum of transmission delay  $d_t$ , propagation delay  $d_p$ , queuing delay in the switches  $d_q$ , and resequencing delay  $d_r$ . We define  $x$  to be the number of hops along a path. With a link capacity  $C$  b/s, a link length  $l_i$  m at hop  $i$ , a buffer size  $B_i$  cells in switch  $i$ , and a velocity of propagation  $v$  m/s, the delay along a path can be expressed as

$$d = \sum d_{t,p,q,r} = ax/C + \sum_{i=1}^x l_i/v + \sum_{i=1}^x aq_i/C + d_r$$

where  $q_i$  is the current queue size, measured in cells, that a cell experiences in switch  $i$  ( $0 \leq q_i < B_i$ ), and  $a = 53 \cdot 8$  is the cell size in bits.

We consider the delay difference among cells from different paths, at the time of arrival at the resequencing buffers. To simplify the calculations, we assume equal buffer sizes of  $B$  cells in the switches, and a constant link length of  $l$  m. We assume that the distance in time between successive cells on a path is preserved during transmission, that is, the delay variation is zero. Thus, the resequencing delay will depend on the largest difference in delay between any two of the paths used for a transmission. We define this delay difference to be  $\Delta d$ . If the paths are of equal length, measured in number of hops,  $\Delta d$  depends only on the queuing delays in the switches. Theoretically, assuming a load close to zero on one path and a load of one on another,  $\Delta d$  would be limited by  $\Delta d < xBa/C$ .

Next, we consider paths of different lengths, but with equal load, and we assume that the difference in queuing delay among the switches on the different paths is negligible. We denote by  $\Delta x$  the largest difference in length, measured in hops, between any two of the paths used for a transmission. With a queue size of  $q$  cells in each switch,  $\Delta d$  can be expressed as

$$\Delta d = (a/C + l/v + qa/C) \cdot \Delta x. \quad (1)$$

Assuming optical fibres, the velocity of propagation is  $v = 2 \cdot 10^8$  m/s. According to (1), with optical fibres and a link capacity  $C = 150$  Mb/s, a one cell increase in switch queue size has as large impact on  $\Delta d$  as a 565 meter increase in link length. For a link capacity of 64 kb/s, the corresponding increase in link length is 1325 km. These results indicate that a large difference in link lengths among the paths corresponds to a small change in load.

In the following we consider the total resequencing buffer capacity required for a transmission over  $N$  paths. The worst case that may occur is when the cells on one path experience a delay  $d_0 + \Delta d$ , while the cells on the

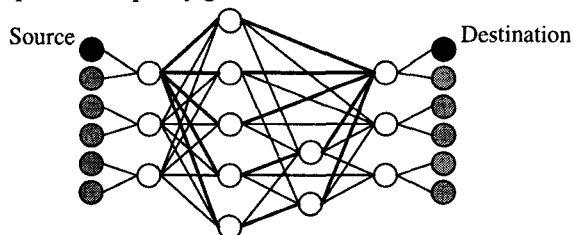
other  $(N - 1)$  paths experience a delay  $d_0$ . Given a source peak rate  $h$  b/s before dispersion, cells are transmitted on the paths at time intervals  $a/h$  s. The sending rate on each path is  $h/(aN)$  cells/s. Thus, during the time  $\Delta d$ , the first path delivers  $\Delta dh/(aN)$  cells to the resequencing buffer, the next path delivers  $(\Delta d - a/h) \cdot h/(aN)$  cells, and path  $i$  delivers  $\delta_i = (\Delta d - a(i - 1)/h) \cdot h/(aN)$  cells to the resequencing buffer. The buffering capacity required in the resequencing buffers, and measured in cells, can then be expressed as

$$B_r = \sum_{i=1}^{N-1} \delta_i = \frac{\Delta dh(N-1)}{aN} - \frac{(N-1)(N-2)}{2N}. \quad (2)$$

For instance, with a delay difference of  $\Delta d = 100$  ms,  $h = 5$  Mb/s and  $N = 5$ , the total required resequencing buffer capacity, according to (2), is  $B_r \approx 942$  cells. By selecting appropriate limits on the delay differences among the paths, the resequencing delay and buffers may be kept at a tolerable level.

## 4.2. Experimental results on resequencing delay

In the following, we present two examples on total delays and resequencing delays in a network. These examples consider non-redundant dispersion over  $N$  paths, and Fig. 8 presents the simulated network topology. When  $N=1$  and  $N=2$ , the paths are of equal length. The differences in path lengths for  $N=5$ , measured in number of hops, were chosen to investigate the effects of dispersion when not all paths are of equal length. When using dispersion, the background traffic in the network is dispersed as well, to obtain the possible capacity gains.



**Fig. 8. Studied network topology.** Each network node is an output-buffered switch with infinite buffer capacity. The measurements are made from the marked source to the marked destination.

In the first example, the link capacity in the network is 150 Mb/s, which corresponds to 155.52 Mb/s links including SDH/SONET overhead [18]. The distance between two adjacent nodes is  $l = 1000$  m, and the velocity of propagation, assuming optical fibre links, is  $2 \cdot 10^8$  m/s. We use output-buffered switches with infinite buffer capacity, in order to study the effects of dispersion without losses. There is a separate FIFO resequencing buffer with infinite capacity for each arriving connection at the destination. In these simulations, we assume that if the cells arrive in order, the resequencing delay is zero. Otherwise, the cells are placed in the buffers, which are served at link capacity. The total simulated time is 10 minutes.

The traffic source used in the simulations is a two-state Markov chain with exponentially distributed sojourn times spent in the *on* and *off* states. The mean on and off times are 0.5 s and 10 s respectively, and while on, the source

generates traffic at constant rate, 150 Mb/s. This corresponds to a mean burst size of 75 Mb, and a source peak-to-mean ratio of 21. The traffic characteristics were chosen to exemplify transmission of medical images [3].

In the second example, we consider voice source parameters, and the mean times on and off are 227 ms and 596 ms respectively [15]. This corresponds to a source peak-to-mean ratio of approximately 4. The source peak rate is  $h = 64$  kb/s, and the link capacity is  $C = 64$  kb/s. The network structure and the rest of the system parameters are as above, and the simulated time is 5 minutes.

Table 1 and Table 2 present the simulation results for the medical image transfer example and the voice source example, respectively. The results show that dispersion in these examples reduces the mean total delay through the network, even though the cells experience resequencing delay at the receiver. Furthermore, the results show that both the mean total delay and the mean resequencing delay for a transmission decrease with increasing dispersion factor. This indicates that dispersion not only reduces the mean delay through the network, but also the variance of the delay. Since the transmission and propagation delays are constant for  $N=1$  and  $N=2$ , and even increase on some paths for  $N=5$ , the reduction is due to reduced queue sizes in the switch buffers. This confirms earlier results on queuing performance [7]. In the simulation model, an increased traffic load is obtained by increasing the mean rate of the sources, hereby reducing the source burstiness. This is a plausible explanation to the reduction in resequencing delay for  $N=2$  and high loads in Table 1. Finally, in the examples considered, the impact of the resequencing delay on the total delay for a transmission in most cases decreases with increasing load and increases with increasing dispersion factor. A plausible explanation to the latter phenomenon is the increased difference in delay among the paths.

In Table 1, for the medical image transfer example, a reduction in mean total delay of up to 90 to 99%, depending on the network load, is obtained with dispersion. The resequencing delay constitutes 1 to 50% of the total delay.

**Table 1. Measured mean total ( $d_{tot}$ ) and resequencing delay ( $d_{res}$ ) for the medical image transfer example.**

$N$	$d_{tot}$ ms (load 0.2)	$d_{res}$ ms (load 0.2)	$d_{tot}$ ms (load 0.5)	$d_{res}$ ms (load 0.5)	$d_{tot}$ ms (load 0.9)	$d_{res}$ ms (load 0.9)
1	105 ± 22	0	367 ± 88	0	1078 ± 109	0
2	5.24 ± 0.9	1.82 ± 0.3	27.4 ± 2.1	6.81 ± 1.0	428 ± 82	4.28 ± 0.70
5	0.16 ± 0.05	0.08 ± 0.04	1.48 ± 0.2	0.64 ± 0.07	137 ± 34	26.2 ± 6.2

Table 2 presents the results from the voice source example, and a reduction of up to 86% is achieved on the mean total delay with dispersion. The resequencing delay constitutes between 10 and 20% of the total delay. Thus, the resequencing delay has smaller impact on the total delay than in the table above. A plausible explanation to this is that the medical image transfer example generates larger queues than the voice source example. Given the difference in number of hops along the paths, larger queues in the nodes result in longer resequencing delays. Furthermore, according to the examples in the previous section, a difference in link length has larger impact on the resequencing delay with link capacity  $C=150$  Mb/s than with  $C=64$  kb/s.

**Table 2. Measured mean total ( $d_{tot}$ ) and resequencing delay ( $d_{res}$ ) for the voice source example.**

$N$	$d_{tot}$ ms (load 0.5)	$d_{res}$ ms (load 0.5)	$d_{tot}$ ms (load 0.9)	$d_{res}$ ms (load 0.9)
1	181 ± 23	0	689 ± 86	0
2	69.8 ± 4.3	10.6 ± 1.3	261 ± 33	30.6 ± 8.7
5	47.1 ± 0.7	5.34 ± 0.1	98.4 ± 3.9	20.9 ± 3.9

Table 3 and Table 4 present results for the same examples as Table 1 and Table 2, but with the link lengths in the network increased to  $l = 100\,000$  m. We notice an increase in total delay due to the increased link lengths, and a smaller increase in resequencing delay. Thus, the ratio of resequencing delay to total delay is not significantly increased. On the contrary, it is decreased for the lowest load in the medical image transfer example. This indicates that in the examples considered, the increased difference in link length among the paths has little impact on the resequencing delay. The same observation was made in Section 4.1.

**Table 3. Measured mean total ( $d_{tot}$ ) and resequencing delay ( $d_{res}$ ) for the medical image transfer example.**

$N$	$d_{tot}$ ms (load 0.2)	$d_{res}$ ms (load 0.2)	$d_{tot}$ ms (load 0.5)	$d_{res}$ ms (load 0.5)	$d_{tot}$ ms (load 0.9)	$d_{res}$ ms (load 0.9)
1	107 ± 22	0	369 ± 88	0	1080 ± 109	0
2	7.22 ± 0.9	1.82 ± 0.3	29.4 ± 2.1	6.81 ± 0.8	430 ± 74	4.28 ± 0.5
5	3.18 ± 0.5	0.38 ± 0.03	4.06 ± 0.08	1.00 ± 0.01	136 ± 24	29.5 ± 3.2

**Table 4. Measured mean total ( $d_{tot}$ ) and resequencing delay ( $d_{res}$ ) for the voice source example.**

$N$	$d_{tot}$ ms (load 0.5)	$d_{res}$ ms (load 0.5)	$d_{tot}$ ms (load 0.9)	$d_{res}$ ms (load 0.9)
1	183 ± 23	0	691 ± 86	0
2	71.8 ± 4.3	10.6 ± 1.3	263 ± 33	32.0 ± 3.0
5	49.8 ± 0.7	5.80 ± 0.3	101 ± 3.8	21.1 ± 1.1

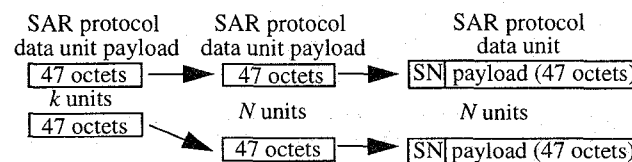
The mean resequencing delay values in Table 1 and Table 2 correspond to mean queue sizes in the resequencing buffers, presented in Table 5. It should be noted that the simulated examples in this study use infinite buffer capacity in the switches. This provides for zero loss, at the expense of long queuing and resequencing delays. A more realistic scenario would be to assume a certain loss, and smaller resequencing queues.

**Table 5. Mean resequencing queue sizes in cells (from the results in Table 1 and Table 2).**

$N$	Medical image (load 0.2)	Medical image (load 0.5)	Medical image (load 0.9)	Voice sources (load 0.5)	Voice sources (load 0.9)
2	644	2409	1514	11	39
5	29	226	9269	7	15

## 5. Adding redundancy

In the case of redundant traffic dispersion, the coding is performed together with the dispersion of the units. We therefore let the SAR sublayer be responsible for the coding and decoding of information. The coding is performed on the payload part of the SAR protocol data units, to encode  $k$  units into  $N$  units, as illustrated in Fig. 9. When the coding is performed, the SAR sublayer adds a sequence number to each of the  $N$  data fields, disperses the units and passes them on to the ATM layer. If the message length is not a multiple of  $k$  units, the last group of dispersed units will consist of the last  $k'$  units from the message, where  $k' < k$ , and redundant units, where the number of redundant units depend on the code being used. The  $k'$  units are sent on paths 1 to  $k'$ , and the redundant units are sent on paths  $k' + 1 \dots N'$ , where  $N' < N$ . The end of a message is indicated by the AUU parameter. When the receiving SAR sublayer is notified by the ATM layer process that an end of message is received on path  $N'$ , it only waits for  $k'$  instead of  $k$  units before decoding. This assumes that the amount of redundancy is constant even in the last subset of cells, that is,  $N - k = N' - k'$ . If the cell which indicates the end of a message is lost, the receiver expects cells from all the  $N$  connections. When these cells do not arrive, the connections are timed out, and the received information is forwarded to the CS. If the number of received units is less than  $k$ , no decoding is performed. When these  $k$  received units do contain encoded information, an erroneous sequence will be delivered to the CS. There are two possibilities to avoid this: (i) not to encode the last  $k'$  units, that is, to set  $N' = k'$ ; (ii) to use a code where  $N' - k' = 1$ . In the latter case, the loss of unit  $N'$  results in loss of the redundant data, and no error is generated in the flow of information delivered to the CS. One code with such properties is the single parity check, which is discussed later in this section.



**Fig. 9. Encoding of information performed at the SAR sublayer:  $k$  units (SAR protocol data unit payload parts), are encoded into  $N$  units. A sequence number (SN) is added to each of the  $N$  units, and the SAR protocol data units are transmitted to the ATM layer.**

The resequencing buffers are polled in the order determined at connection establishment to read out a correct sequence of units. If a unit is lost or severely delayed on one connection, and there is no redundancy available to reconstruct it, the resequencing must proceed without it. When a unit is received on one connection, there should be a maximum time to wait for a unit on the next connection, before considering it lost. The time limit must be short enough not to cause severe performance degradation due to long resequencing delays. When a traffic contract has been established, specifying the cell transfer delay and the cell delay variation [20], these parameters can be used to decide when to time out a cell. If a unit does arrive, but with higher sequence number, the cell that did not arrive is known

for certain to be lost, and the resequencing proceeds without it.

An encoded part of a message which is transmitted in  $N$  units should be possible to reconstruct using any  $k$  units thereof. A group of codes with these properties is the Reed-Solomon codes, or, if  $N - k = 1$ , the single parity check [2]. We propose the AAL to support the single parity check code, which is applicable for all  $N > 1$  through a simple exclusive-OR calculation. Furthermore, the single parity check can encode the last units of a message, even though the number of units in that group is not equal to  $k$ .

The SAR process at the receiving side polls the resequencing buffers of the terminating connections in a given order until  $k$  units have been collected. The coding hence enables the reconstruction of missing units. Since the units are uniquely defined by the sequence numbers, a missing unit that was merely late and not lost can be purged when it finally arrives. As soon as any  $k$  of the  $N$  units arrive at the SAR sublayer, they are resequenced, reassembled and passed on to the CS.

Assume that the probability of cell loss on a path is  $\phi$ . When using disjoint paths, losses on different paths due to transmission faults are independent. Hence, the probability of  $x$  losses in a group of  $N$  cells, dispersed over  $N$  paths, is binomially distributed according to

$$Pr\{x \text{ losses}\} = \binom{N}{x} \phi^x (1 - \phi)^{N-x}.$$

With a single parity check, one loss in each group of  $N$  dispersed cell can be corrected. The probability of loss after decoding at the receiver can thus be expressed as

$$Pr\{x \geq 2\} = 1 - (1 - \phi)^N - N\phi(1 - \phi)^{N-1}. \quad (3)$$

Fig. 10 shows the loss probability according to (3) as a function of the loss probability on the paths,  $\phi$ .

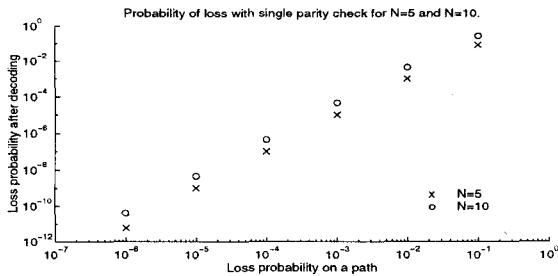


Fig. 10. The probability of loss after decoding as a function of the loss probability on the paths. The code is a single parity check and the graph shows the loss probability for different degrees of dispersion.

The single parity check reduces the loss probability from  $10^{-6}$  to  $10^{-11}$  or from  $10^{-5}$  to  $10^{-9}$ , for a dispersion factor  $N = 5$ . The coding thus improves the loss performance at the expense of increased capacity, with an overhead of  $N-k$  cells. However, dispersion was in [5] shown to reduce the required capacity for a transmission, and adding a single parity code may in many cases still require less capacity than an ordinary non-dispersed and non-redundant transmission.

## 6. Pseudocode for receiving AAL 5'

Lastly, to illustrate the complexity introduced in an AAL supporting traffic dispersion, we present pseudocode for resequencing and decoding of information units at the receiving AAL. The code in Fig. 11 assumes separate FIFO resequencing buffers and sequence numbering as described in Section 4. In this example, we put any  $k$  of  $N$  arrived units in a vector and decode them to obtain the  $k$  original units. The decoding function is not explicitly described.

```

count = 0 // to poll buffers in cyclic order; modulo N
buffer[0]..buffer[N-1] // resequencing buffers
vector[0]..vector[N-1] // vector for decoding
seq_nr = 0 // sequence number for N units ; modulo 256
group_size = k // number of non-redundant units in a group

number in vector = 0 // starting with an empty decoding vector
start timer // to check for late or lost units

while !end of message
  while number in vector < group_size
    check first unit in buffer[count]
    if sequence number = seq_nr
      put the unit in vector[count]
      number in vector = number in vector + 1
      restart timer
    else
      if sequence number < seq_nr // modulo 256;
        late arrival discovered
        discard cell
      if sequence number > seq_nr // modulo 256
        block the buffer // do not try to read this
        buffer until seq_nr has changed
      if vector[count] is empty // loss discovered
        mark vector[count] empty
        number in vector = number in vector + 1
        restart timer
      if timer > limit
        mark vector[count] empty
        number in vector = number in vector + 1
        restart timer
    if end of message reached
      set group_size // may not await k cells in the last group
      count = count + 1 // modulo N
      decode vector // exclusive-OR for single parity check
      forward SAR protocol data units from decoded
        vector[0]..vector[group_size-1] to CS
      seq_nr = seq_nr + 1 // modulo 256
      empty vector
      number in vector = 0

```

Fig. 11. Pseudocode for resequencing and decoding of dispersed units at receiving AAL.

## 7. Concluding remarks

In this study, we have discussed the protocol functions necessary to support traffic dispersion in an ATM network. We suggest that the network decides what degree of dispersion to employ for each call. If the user explicitly requests dispersion, it should be stated in the AAL parameters in the SETUP message. The connection setup is performed with a call reference common to all the virtual connections used for a transmission, to minimize the number of signalling messages.

Earlier results have shown a general tendency for benefits due to dispersion to increase with increasing dispersion

factor [5][7]. However, there may be practical limitations to finding a large number of disjoint paths of similar lengths through a network. We propose that the maximum dispersion factor to be supported by AAL type 5' is set to 10. This suggestion is supported by the results in [5][7], which show that in general, a dispersion factor between two and ten appears sufficient to obtain most of the possible queuing and capacity gains due to dispersion. Consequently, we suggest that a dispersion factor  $N \in [1, 10]$  is selected. The choice of dispersion factor for a call is based on the number of available paths in the network, the source peak-to-link ratio and, if known, the source peak-to-mean ratio. As discussed in this study, the differences in load and length among the paths should also be small to minimize the resequencing delay.

Considering the traffic classes defined by ATM Forum [20], traffic dispersion may be used for reliability or security reasons for constant bit-rate traffic (CBR), or to provide more capacity than a single link can offer. For variable bit-rate (VBR), available bit-rate (ABR) and unspecified bit-rate (UBR) traffic, dispersion may also improve the performance of statistical multiplexing and reduce the number of retransmissions by using forward error correction. ABR traffic provides a possibility for the network to continuously change the maximum allowed transmission rate for a source [1]. This requires further refinement of some protocol functions for traffic dispersion. Another topic which remains for future work is protocol functions supporting multicast with traffic dispersion.

We have discussed how to structure the SAR sublayer and the ATM layer processes, how to introduce coding and how to perform resequencing. To be able to resequence cells from different connections at the receiver, we introduced sequence numbers in AAL type 5'. We suggest the AAL type 5' to support a single parity check coding. With redundant dispersion over five paths, a single parity check was shown to improve the cell-loss probability from  $10^{-5}$  to  $10^{-9}$ , provided that losses on different paths are independent. Resequencing is suggested to be performed at the receiving SAR sublayer with physically or logically separate FIFO resequencing buffers for the arriving virtual connections. We also presented a few examples on cell transfer delay through a network, with and without dispersion. Even though the cells suffered resequencing delay at the receiver, dispersion reduced the total delay in these examples. Lastly, the increased complexity of the AAL due to dispersion was illustrated through a pseudocode example.

In summary, the overhead introduced by dispersion is  $4(N - 1)$  octets in the signalling messages defining VPCI/VCI values for a call,  $N - 1$  additional ATM layer processes for each call, encoding and decoding in the case of redundancy, and resequencing at the receiver. There is also a 2% overhead per cell due to cell sequence numbers when using AAL type 5', and routing overhead due to the need of finding multiple paths for each call. In return, dispersion has been shown to decrease the capacity requirements [5], to decrease the network delay despite resequencing in the examples considered, and to increase the network security and tolerance to faults. We therefore believe traffic dispersion to be a competitive strategy when dealing with unpredictable and strongly correlated traffic in ATM networks.

## 8. Acknowledgements

The simulations were performed on the sequential simulator YESS, developed by the Simulation Laboratory at KTH Dept. of Teleinformatics. The authors especially thank Robert Rönngren and Mikael Schulz for their valuable help and comments.

## 9. References

- [1] The ATM Forum Technical Committee: ATM User-Network Interface (UNI) Signalling Specification, Version 4.0, af-sig-0061.000, July 1996.
- [2] V.K. Bhargava: "Forward error correction schemes for digital communications", IEEE Comm. Magazine, Vol. 21, No. 1, January 1983, pp. 11-19.
- [3] J. E. Cabral Jr., Y. Kim: "Multimedia Systems for Telemedicine and Their Communications Requirements", IEEE Comm. Magazine, Vol. 32, No. 7, July 1996, pp. 20-23.
- [4] CCITT Recommendation I.321, B-ISDN Protocol Reference Model and Its Application, 1991.
- [5] E. Gustafsson, G. Karlsson: "When Is Traffic Dispersion Useful? A Study on Equivalent Capacity", Performance Modelling and Evaluation of ATM Networks, Second volume, Ed. D. Kouvatso, Chapman & Hall, 1996, pp. 110-129.
- [6] E. Gustafsson, G. Karlsson: "A Literature Survey on Traffic Dispersion", IEEE Network, Vol. 11, No. 2, March/April 1997, pp. 28-36.
- [7] E. Gustafsson, G. Karlsson: "The Strategy of Traffic Dispersion", Proc. of the Seventh IFIP Conference on High-Performance Networking, HPN'97, White Plains, New York, USA, May 1997, pp. 280-294.
- [8] ITU-T Recommendation I.311, B-ISDN general network aspects, 1993.
- [9] ITU-T Recommendation I.361, B-ISDN ATM layer specification, 1993.
- [10] ITU-T Recommendation I.362, B-ISDN ATM Adaptation Layer (AAL) Functional Description, 1993.
- [11] ITU-T Recommendation I.363, B-ISDN ATM Adaptation Layer (AAL) Specification, 1993.
- [12] ITU-T Recommendation I.371, Traffic Control and Congestion Control in B-ISDN, 1993.
- [13] ITU-T Recommendation I.432, B-ISDN User-Network Interface - Physical Layer Specification, 1993.
- [14] ITU-T Recommendation Q.2931, Broadband Integrated Services Digital Network (B-ISDN) - Digital Subscriber Signalling System No 2 (DSS2) - User-Network Interface (UNI), Layer 3 Specification for Basic Call/Connection Control, 1995.
- [15] H.H. Lee and C.K. Un: "A Study of On-Off Characteristics of Conversational Speech", IEEE Trans. on Communications, Vol. COM-34, No. 6, June 1986, pp. 630-637.
- [16] W.E. Leland et al.: "On the Self-Similar Nature of Ethernet Traffic (Extended Version)", IEEE/ACM Trans. on Networking, Vol. 2, No. 1, February 1994, pp. 1-15.
- [17] V. Paxson, S. Floyd: "Wide-Area Traffic: The Failure of Poisson Modeling", IEEE/ACM Trans. on Networking, Vol. 3, No. 3, June 1995, pp. 226-244.
- [18] S.K. Rao and M. Hatamian: "The ATM Physical Layer", ACM Computer Comm. Review, Vol. 25, No. 2, April 1995, pp. 73-81.
- [19] J.M. Simmons and R.G. Gallager: "Design of Error Detection Scheme for Class C Service in ATM", IEEE/ACM Trans. on Networking, Vol. 2, No. 1, February 1994, pp. 80-88.
- [20] K-Y. Siu and R. Jain: "A Brief Overview of ATM: Protocol Layers, LAN Emulation, and Traffic Management", ACM Computer Comm. Review, Vol. 25, No. 2, April 1995, pp. 6-20.
- [21] B. Stiller: "A Survey of UNI Signalling Systems and Protocols for ATM Networks", ACM Computer Comm. Review, Vol. 25, No. 2, April 1995, pp. 21-33.