PAPER

Forward Error Correction Control on AAL 5 : FEC-SSCS

Kumiko Kanai† **Keiji Tsunoda** † **Shigeo Matsuzawa** † ***and*** **Hiroshi Esaki** † ***Members***

SUMMARY A novel cell-level FEC (forward error correction) scheme at SSCS of AAL type 5 for error-free data transmission services in ATM networks is proposed and evaluated. In the proposed cell-level FEC scheme, both the length of user data (e.g., IP packet) and attached redundant data can be modified based on sender's local decision without any end-to-end parameter re-negotiation procedure. The writing and reading order regarding an interleave matrix for cell-level FEC algorithm are the same, in order to perform pipelining data transmission. The prototype implementation with software processing achieves few Mbps end-to-end throughput.

The end-to-end data transmission latency and the amount of retransmission packets due to packet error at the receiver entity are evaluated by computer simulation with correlated cell dropping process. Simulation results show that the benefits of cell-level FEC scheme for the error-free data transmission services, i.e., by use of cell-level FEC scheme, the amount of retransmitted packets can be reduced, even if the average latency for end-to-end data transmission increases slightly.
key words: ATM, FEC, reliable data transmission

1. Introduction

An ATM network is realized as the future communication platform that can accommodate variety of communication services, that include realtime data (e.g., voice or video) and non-realtime data (e.g., electronic mail). Some communication services require end-to-end error-free data delivery and others do not. In order to provide end-to-end error-free data delivery, some error control has to be performed. For example, in TCP/IP protocol suite, TCP provides error-free data transmission by using packet retransmission. Latency and throughput of end-to-end data delivery are the most important performance measurements for communication services requiring error-free data delivery.

ATM network is a kind of packet network, and all user data is transferred by 53 Bytes fixed size packet, called cell. User data (e.g., IP packet) has to be fragmented into multiple cells in ATM networks. Since the buffer and bandwidth resources in ATM networks will be statistically shared among multiple data flows, some cells may be dropped due to buffer overflow. As briefly discussed in section 2, some applications can not achieve sufficient quality for end-to-end data delivery, due to cell loss in ATM networks [1]. Especially both for the communication with large delay and bandwidth product (i.e., for long haul data flow) and for the communication with large sized packet, performance degradation due to cell loss will be serious.

In order to improve error characteristics for data delivery over ATM networks, we propose a novel forward error control (FEC) scheme over AAL type 5 [2] [3] [4]. The proposed FEC scheme called FEC-SSCS in this paper is cell-level at SSCS (Service Specific Convergence Sublayer). Also, the proposed FEC scheme does not require any modification in SAR nor CPCS of existing AAL type 5. In other words, the proposed FEC-SSCS sits on the existing SAR/CPCS of AAL type 5.

The benefit of FEC-SSCS for error-free data communication services is shown by the performance evaluation with computer simulation. The applications, that would have a benefit to apply to the FEC-SSCS, could be the large scale reliable multicast applications (e.g., discussed in [5]) and would be the interactive game applications. These applications require some delay bound, however, those applications does not always (or continuously) transfer the data as assumed in FEC options in I.363 [6].

Section2 discusses issues in conventional error control schemes over ATM networks. Section 3 discusses the specification of FEC-SSCS. Section 4 shows performance evaluations of proposed FEC-SSCS by computer simulation, and discusses the prototype implementation. Finally, section 5 gives a brief conclusion.

2. Issues of Error-Free Data Transmission over ATM Networks

In ATM networks, AAL type 5 [6] will be the most commonly available AAL protocol for many kinds of data transmission services. AAL type 5 only detects error in AAL5 service data-units, which are represented as CPCS-SDUs, and never carry out error correction for them. Since neither ATM nor PHY layer has error correction capability for user data, ATM networks with the conventional AAL type 5 provide only error detection for CPCS-SDUs. Errors of received CPCS-SDUs will occur due to cell loss and/or bit error(s). As the error detection is performed for whole of CPCS-SDU, it is impossible to know which cell is dropped or which cell

Manuscript received February 25, 1998.
Manuscript revised May 14, 1998.
†The authors are with R&D Center, TOSHIBA CORPORATION, Kawasaki-shi, 210 Japan.

includes bit error(s). Therefore, whole of CPCS-SDU that corresponds to multiple cells will be dropped, even if the received CPCS-SDU experiences only one bit error or only one cell loss. Moreover, since existing transport protocols (e.g., TCP) use packet retransmission for recovery of erroneous data packets, the whole of erroneous packet has to be retransmitted from the sender, even when the number of dropped or erroneous cells is small and many cells are correctly transmitted to the destination. This is very serious for high speed data transmission services, for the high speed data transmission service with small latency requirement, that generally use a large packet size to reduce the number of transmitted packets (i.e., reducing the overhead contributed by packet header transmission and the frequency of context switching at receiver hosts according to packet reception), and for large scale error-free multicast data transmission [1]. For example, for a datagram of 64 kB($\approx$ 1330 cells), packet error probability is expected to be about 5×10^{-4} for uncorrelated errors with a bit error rate (BER) of 10^{-9}, and is expected to be about 0.6% for uncorrelated cell loss with a cell loss ratio of 10^{-6}. For packet of 9180 bytes, that is the size proposed as a default ATM-MTU [7], and the packet error probability is expected to be about 2×10^{-4} for uncorrelated cell loss with a cell loss of 10^{-6}. Moreover, in the case of reliable multicast services, sufficient performance in throughput and latency cannot be expected, since packet error due to bit errors will linearly increase with the number of receivers.

The other issue is performance degradation due to an unexpected congestion in ATM networks. Since the communication resources (i.e., bandwidth and buffer) are statistically shared among multiple data flows, cell transmission service quality relies on the co-operation of each entity (i.e., end-station and switch node). This is especially true for ABR services. Although UPC (Usage Parameter Control) and NPC (Network Parameter Control) are defined, as functions that monitor and police each cell flow they are not mandatory function and will not be enough to control the cell flows. Therefore, even when we have UPC/NPC, unexpected congestion will occur both due to mis-behavior of nodes (end-stations and switch nodes) and due to the statistical traffic fluctuation.

Yet another issue for error-free data transmission, that is not ATM specific, is poor data transmission performance for packet flow with a large delay and bandwidth product. This is because commonly used transport protocols (e.g., TCP) apply dynamic window control policy. In order to achieve sufficient throughput, the window size has to be large enough to perform pipelining packet transmission. However, once packets are erroneous, the window size will be quickly shrunk and will increase slowly. When the window size is small, packets can not be transmitted in pipelining and can not achieve high throughput. Therefore, when the packet error ratio becomes large, data transmission performance is significantly degraded. As discussed above, the packet error probability in ATM networks will be large for large size packets. Then, in ATM networks, we could not achieve sufficient throughput for the packet flows with a large delay and bandwidth product.

As related works, several FEC schemes has been proposed for cell discard [6] [8] [9] [10]. [10] describes a modified Reed-Solomon code, which simplifies error correction algorithm. [9] describes a two dementiponal interleaving FEC frame structure, in order to deal with bursty cell loss. [8] gives some performance evaluation with video and bursty sources and shows the effectiveness of FEC scheme. However, in these papers, a fixed-size FEC frame is suggested, and the re-ordering of data transmission is required to deal with bursty cell loss. Therefore, when we transfer the variable sized packets with these methods, the end-to-end latency increases significantly due to the transmission of unnecessary cells and due to the re-ordering of transmission data. Moreover, these schemes can not modify the length of appended data, that determines the error correction capability. In ATM networks, the cell loss ratio and cell loss pattern is different for each service class and for each QoS class. And, also, the end-to-end service quality required by the application is not always the same during a session. As a result, in many cases, the FEC scheme with fixed length of appended data will not be effective or will require large and unnecessary large transmission overhead. In ITU-T I.363.1 FEC option for CBR flows, it is assumes that the user data is always and continuously transferred from the source node to the destination nodes. However, in many other applications, the sender nodes do not always (or continueously) transfer the data to the destination nodes.

In the next section, we propose the FEC scheme that can solve the issues mentioned above. The proposed FEC scheme can effectively transmit variable length packets and can easily modify the appended data length even during a session.

3. Specification and Implementation of Cell-Level FEC Scheme : FEC-SSCS

In this section, we discuss the specification of FEC-SSCS protocol and it's prototype system implementation.

3.1 Necessity of FEC Control at AAL Layer

As discussed in the previous section, error-free data transmission over ATM networks will not be able to achieve sufficient end-to-end performance in some cases, e.g., with large size packet. In order to solve this issue, we must reduce packet error probability at the receiver entity. FEC, forward error correction, is the most likely solution to reduce the packet error probability.

FEC scheme could be applied to PHY or application level, as well as AAL level proposed in this paper. The FEC scheme at application level will work well. However, we can not expect high throughput by this approach. Also, since the FEC scheme is packet level instead of cell level, latency for error recovery, when the FEC scheme can not recover the correct data, will be significantly larger than AAL-level FEC scheme. The FEC scheme at PHY level can only correct bit errors on the transmission medium, and therefore can not provide cell loss recovery.

When we apply an FEC scheme at AAL level, packet error both due to cell loss and due to bit error can be recovered and we need not modify the upper layer applications such as TCP/IP. Also, since the error recovery is performed within the packet, the error recovery latency, when the FEC can not get a correct data, will be much smaller than that for FEC scheme at application level. Finally, when the AAL-level FEC scheme is implemented at the interface board, we can provide high throughput data delivery to the application. Here, the AAL-level FEC scheme can be implemented within the operating system at application layer. In this case, we could not expect high throughput data communication. However, still, the error recovery latency, when the FEC can not get a correct data, will be much smaller than for FEC scheme at packet level. Therefore, as also discussed in [2], an AAL-level FEC scheme is necessary for the applications requiring some delay bound for end-to-end data transmission.

In the following subsections, we discuss requirements and specification of AAL-level FEC scheme at service specific convergence sublayer (SSCS).

3.2 Requirements of FEC-SSCS

The followings are requirements and design goals of the AAL-level FEC scheme.

- The FEC-SSCS should have a compatibility with the specification of the existing AAL5, i.e. compatibility with the current SAR/CPCS for the AAL5. This is because not all applications apply FEC-SSCS, and therefore conventional SAR/CPCS can be commonly used either with or without FEC scheme.
- Variable size of data, that is different assumption from ITU-T I.363.1 [6] does, should be allowed. However, transmission overhead of FEC-SSCS should be minimized.
- The amount of redundant data should be minimized to get small latency and high throughput.
- The amount of appended redundant data should be able to be optimized according to the network status and packet length with small control overhead. Optimization of the redundant data length reduces latency and increases throughput.
- It should be possible to adjust and negotiate parameters of the FEC algorithm. These parameters should be able to be negotiated during a session as well as connection establishment phase.

- FEC-SSCS should provide three operational correction modes(bit error correction, cell loss correction and bit error/cell loss correction) with the same algorithms according to the network conditions. (We will assume only the case cell loss correction is provided for performance evaluation)
- The data processing cost at the receiving entity should be minimized when there is no error in the received packet. In this case, the data processing speed is almost the same as existing the AAL5.
- In order to avoid increase of latency, packets should be transferred in pipelining. Also, by the pipelining, the required buffer space at the sender entity can be reduced.

The proposed FEC scheme satisfies all requirements and design goals listed above. In the following subsection, we only describes the functional behavior of sender and receiver entities. The detailed specification of the proposed FEC scheme is described in [3].

3.3 Behavior of Sender and Receiver Entities

Figure 1 gives functional abstraction and data formats of the FEC-SSCS. Each FEC frame has two demensional structure, $M \times (N + n)$ symbols. FEC frame contains two parts. One is a data-part ($M \times N$ symbols) that contains user data, and other is a code-part ($M \times n$ symbols) that contains redundant data for FEC. All parameters such as M, N and n are negotiable at connection establishment phase.

M, that is usually 46 Bytes (=cell payload length $-$ SSCS header), is length of vertical line in the FEC frame. N, that depends on the bit length of symbol defined in the FEC algorithm, is the maximum number of vertical lines in the data-part. n is the maximum number of vertical lines in the code-part.

Sender side behavior

1. The parameter negotiation is performed through user-to-user signaling provided by Q.2931 or through in-band signaling.
2. Each AAL-IDU (e.g., IP packet) is written in the data-part of an FEC frame as shown in Figure 1. When the AAL-IDU is larger than the data-part

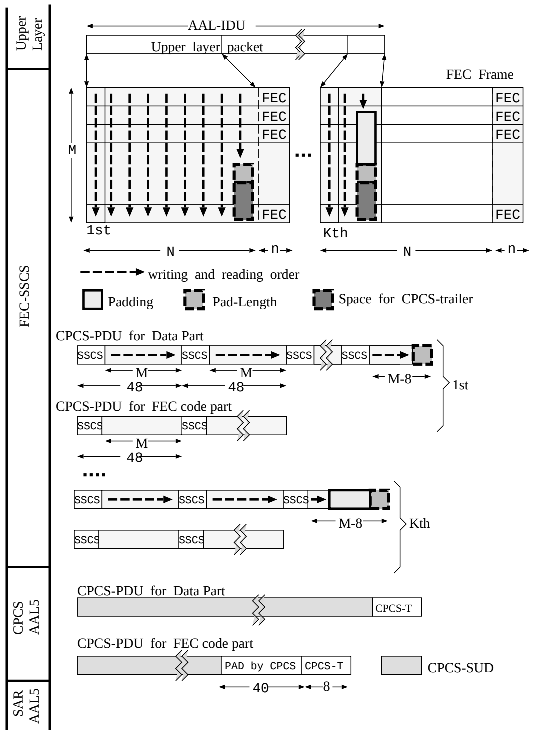

Figure 1. FEC-SSCS process

of FEC frame size, it is divided into multiple FEC frames. User data is written and read in pipelining, without any data re-ordering. Also, as described in [3], appended FEC code can be calculated in pipelining. Therefore, whole of data-part does not need to be stored at the sender entity.

Both for AAL-IDUs that are smaller than the data-part of the FEC frame and for the last part of AAL-IDU that is also smaller than the data-part of the FEC frame, whole of the data-part in the FEC frame is not filled with the user data. The data-part, that is not filled with user data, is assumed to have some pre-defined symbol pattern to calculate FEC redundant data (i.e., code part symbols), and does not need to be transmitted to the receiver entity. Here, the predefined symbol pattern for the data-part has to be shared between sender and receiver entities. By this operation we can avoid sending unnecessary data, that may cause degradation of cell transmission efficiency.

3. After the step 2, the data in code-part is read out. Since writing and reading order for interleave matrix are the same, the source FEC-SSCS entity can transmit SSCS-PDU in pipelining.
4. When the data is read out from the FEC frame, SSCS headers are attached to each M byte length data. The SSCS header has, at least, an SN (Sequence Number) and a CRC-10 check sequence field. The SN field identifies which cell is dropped at the receiver entity. The CRC-10 field detects upto three bit errors in each vertical line. A single bit error in the vertical line may be corrected using the CRC-10.
5. Each FEC frame is divided into two CPCS-PDUs corresponding to the data-part and the code-part. In order to minimize the number of transmitted cells for data-part, the last 8 bytes in the data-part field is reserved for CPCS trailer. Regarding transmission of code-part, 40 bytes of padding is required at CPCS since the code-part has a 48 bytes alignment. This means that the last cell transmission of code-part has no effective data and contains padding information only.

   Though [3] describes an alternative method to map FEC frame and CPCS-SDU that avoids transmission of 40 byte padded cell, the above mentioned method is assumed in this paper. Since it makes functional behavior at the receiver entity simpler.
6. The data is given to the CPCS process.

<u>**Receiver side behavior**</u>

1. The CPCS process is performed.
2. In the case that CPCS layer detects some errors by CRC-32, indication of that errors is reported to the upper layer(FEC-SSCS) by the CPCS. Here, when the error for the received data is not detected, the received data can be transferred to the upper layer without data re-ordering. This is because the writing and reading order of user data are always the same for end-to-end data transmission.
3. When the data-part has at least one error, the data-part and the code-part have to be restored the correct position in the FEC frame. And, the CRC-10 in SSCS header detects which line is erroneous due to bit error(s). The erroneous vertical line(s) of FEC frame is(are) assumed as if the erroneous vertical line(s) is(are) missing. Using the information of which vertical line(s) are missing or erroneous, the FEC algorithm tries to recover the original data-part information.
4. When only the code-part is erroneous, no action is performed.
5. SSCS headers, Padding, pad-length fields are removed and the error-free data is provided to the upper layer.

4. Performance Evaluation of FEC-SSCS

In this section, the performance of proposed FEC scheme is evaluated by computer simulation, and is evaluated by prototype implementation.

4.1 Simulation Model and Parameters

Protocol and Architecture Model

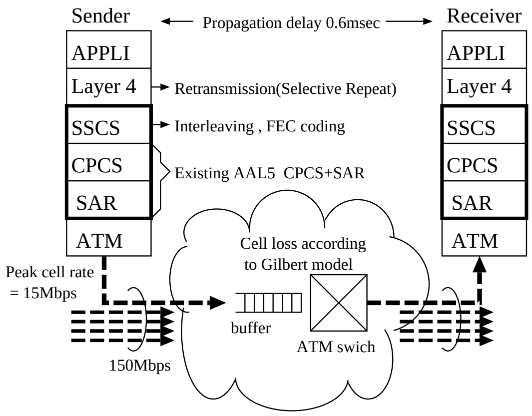

Figure 2. Simulation Model

Figure 2 shows a simulation model. Packets are transferred from a Sender to a Receiver. Propagation delay, including cell queuing delay at switch nodes, between the Sender and the Receiver is 0.6 *msec*. 0.6 *msec* corresponds to the propagation delay for about 180km or corresponds to the queuing delay for about 222 cells with 155Mbps link. The Sender transmits data packets whose size are constant (i.e., 1, 4, 8 and 64kBytes in the simulation). An available peak rate for the Sender is 15Mbps.

FEC processing time at the Sender can be assumed as negligible, due to pipelined data transmission and encoding of the appended data. At the receiver, following FEC processing time for error recovery and that for CPCS trailer check is assumed.

CPCS trailer check: 2.7 μsec
$\rightarrow$ (1 cell transmission time for 155Mbps link)
FEC16: 0.0432 ms/frame
$\rightarrow$ (16 cell transmission time for 155Mbps link)
FEC128: 0.3456 ms/frame
$\rightarrow$ (126 cell transmission time for 155Mbps link)

The additional processing delay generated at the layers (e.g., APPLI, Layer 4, SAR and ATM in Figure 2) is assumed null in this simulation. Since the value of it depends on the implementation and is common for FEC scheme and noFEC scheme, we don't need to consider the value for the comparison. In the case that no error is detected, the FEC process is not invoked and no FEC processing time is required at FEC-SSCS entity.

Both with and without the FEC scheme, the upper layer (i.e., Layer 4 in Figure 2) performs error correction based on selective repeat with a static window control, whose window size is one. With the FEC scheme, only packets that can not be recovered by the FEC scheme are selectively retransmitted.

In the ATM cloud, only cell loss is assumed to occur. We use a correlated cell discarding process with "a two-state Gilbert model", as shown in figure 3. Cells are consecutively dropped in Bad state and no cell is dropped in Good state [9]. The average number of consecutive cell loss is given by $1/(1-q)$, and the average cell loss ratio is given by $(1-p)/(2-p-q)$. We use $q = 0.1$ and $q = 0.5$ in the evaluation.

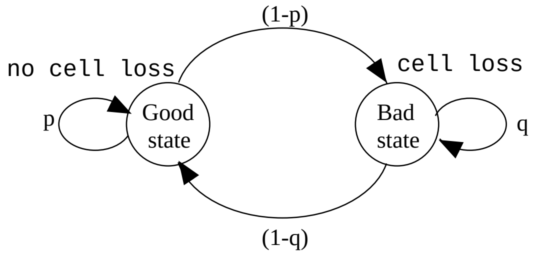

Figure 3. Cell Discarding Process Model

Two types of FEC frame are evaluated; one is FEC(16,15) which signifies data-part=15cells, code-part=1cell, and the other is FEC(128,124) which signifies data-part=128cells, code-part=4cells. When the number of lost cells in an FEC frame is larger than the number of cell in FEC code-part, the FEC scheme can not restore the original data and the packet must be retransmitted.

Performance of the FEC scheme is evaluated in terms of *Latency* and *Efficiency*, whose definitions are:

$$\begin{aligned} &\textit{Latency} \\ &\quad = \text{moment to complete data reception} \\ &\qquad\qquad - \text{moment to start data transmission} \end{aligned}$$

$$\begin{aligned} &\textit{Efficiency} \\ &\quad = \frac{\text{amount of received AAL-IDU}}{\text{amount of cells sent to the network}} \end{aligned}$$

Latency is a required time period from the moment that head of the packet is transferred from the application to AAL entity at the sender to the moment that tail of the packet is transfered to the application at the receiver. *Efficiency* is the ratio of the amount of transferred data between the sender and the receiver at an application level to the amount of actually transmitted cells (including redundant data and retransmitted data) from the sender to the receiver.

The characteristics of *Latency* is shown in figures 4 - 9. Increase of the *Latency* with FEC scheme, compared with the *Latency* without FEC scheme is small. Without FEC scheme, the average *Latency* dramatically increases in accordance with the increase of cell loss ratio. Especially, for 64kBytes packet size, the average *Latency* becomes significantly large when the cell loss ratio is about 10^{-4}. On the other hand, with FEC scheme, the average *Latency* is not that large though the cell loss ratio is about 10^{-3}. For 1 kByte packet, the average *Latency* increases slightly, even when the cell loss ratio is about 10^{-2}; Figures 4 and 7.

For stronger correlated cell loss (i.e., larger q in the simulation), we need larger amount of redundant cells. Therefore, as shown in the Figure 4 - 9, the *Latency* for FEC128 which has for redundant cells is better than for FEC16 which has only one redundant cell. When we use larger number of redundant cells, we can obtain better *Latency* characteristics even at large cell loss ratio.

For many applications, delay jitter is much important than the average delay (i.e., latency). In order words, many applications are not sensitive to a slight increase of average delay, but are sensitive to an increase of delay jitter. Though we have not shown the evaluation results with regard to delay jitter, simulation results show that the FEC scheme can reduce the variance of latency.

Figures 10 - 15 show the characteristics of *Efficiency*. Degradation of *Efficiency* with FEC scheme due to packet retransmission can be small compared with the case that the FEC scheme is not applied to. Without FEC scheme, the *Efficiency* degrades dramatically in accordance with the increase of cell loss ratio. Especially, for 64kBytes packet size, the *Efficiency* significantly degrades for about 10^{-4} cell loss ratio. On the other hand, for the FEC128, degradation of the *Efficiency* due to the packet retransmission can be small even when the cell loss ratio is about 10^{-3}. For 1 kByte packet, degradation of the *Efficiency* due to packet retransmission is small even when the cell loss ratio is about 10^{-2}.(Figures 10 and 13)

Efficiency is an important performance parameter. However, from the view point of error-free data transmission, probability of packet retransmission would generally be more important performance parameter. This is because the window size will shrink and the packet transmission rate from the sender process will be reduced due to the packet retransmission.

Since the window size shrinks to small value quickly and increase it slowly with commonly available window control policy, end-to-end throughput will degrade according to the increase of packet retransmission events.

The proposed FEC scheme will improve the end-to-end error-free data delivery performance with slight increase of average latency, though it requires an additional FEC processing delay at the receiver entity to correct the erroneous data and a transmission overhead due to the appended redundant data.

How many redundant data per FEC frame is needed in order to obtain sufficient performance mainly depends on the cell loss pattern. Larger number of redundant data is required for the strongly correlated cell loss. On the contrary, the size of FEC frame may be determined based on the end-to-end latency requirement by the application and based on the packet size generated by the application. As mentioned in section 3, the proposed FEC scheme can easily optimize both the number of redundant data per FEC frame and the (actual) FEC frame size even during a session. As a result, the proposed FEC scheme will be able to achieve sufficient throughput and latency performance with reasonable transmission overhead.

4.2 Prototype System Implementation

4.2.1 System Configuration

The auhtors implement the pproposed FEC-SSCS mechanism using an UNIX system. FEC processing is implemanted as the kernel process, and the conventional SOCKET interface is used as the interface between the kernal and application process.

The followings are the system configuration for prototype system.

- OS; BSD/OS Version 3.0
- CPU; Pentium 166MHz with 48MB memory
- FEC algorithm ; Reed Solomon Block Erasure [10]
  - Generator polynomial; $G(x) = x^8 + x^4 + x^3 + x^2 + 1$
  - Symbol; 8 bits
  - FEC frame; M=47Bytes, (N+n)≦ 255, n≦ 4

4.2.2 Performence Evaluation of Prototype System

The FEC-SSCS process is executed purely by software processing, without any special hardware for FEC-SSCE processing. The application, applying the FEC-SSCS mechanism, is using the UDP socket to transfer the data. Between the source and destination hosts, there is no router. This means that source and destination hosts exist on the same link .

Tables 1 shows the actual end-to-end data throughput at the UDP socket. 8KBytes size IP packtes are transferred from the source host to the destination host. FEC frame size is defined so as to fit with 8KBytes data size. Therefore, an FEC frame corresponds to 172 cells. The number of redundant cells attached to each FEC frame can be upto four.

The end-to-end throughput can be larger than 3Mbps, even when four cells in an FEC frame (8KBytes) are dropped in the network. This is the case

Table 1. End-to-end Throughput with Prototype System

Redundant cells	Without cell loss	With cell loss
1	18.8 Mbps	5.7 Mbps
2	13.5 Mbps	4.8 Mbps
3	9.9 Mbps	4.3 Mbps
4	7.4 Mbps	3.6 Mbps

where the FEC process is executed purely by software, without any special hardware processing.

Many existing applications, including multimedia applications, could accept the few Mbps end-to-end throughput. For example, with few Mbps throughput, we can transfer the MPEG1 stream with a real-time transmission. This means that, even with software implementation of FEC-SSCS, many applications will be able to use the FEC-SSCS mechanism, in order to improve the end-to-end data transmission quality.

4.3 Discussion

The evaluations performed in this paper are for between the end-hosts, that both hosts are directly connected through the same ATM network. However, in the real network, it would be hard to assume that the source and destination hosts sit on the same ATM network and that the source and destination hosts can have a direct ATM connection between them. In other words, in many cases, we will have some intermediate node(s), called as a router, between source and destination hosts. When we have any intermediate router between source and destination nodes, the ATM connection has to be terminated at the intermediate node. There would be many ways, how to handle the received IP packet flows received by the intermediate node. The intermediate node could aggregate the different IP flows in an ATM connection, that is shared by multiple IP flows. Or, the intermediate node could use a dedicated ATM connection for a particular IP flow between source and destination nodes. This means that, an intermediate node in the ATM network will have multiple ATM connections, that would be terminated at the other intermediate nodes or at the end-hosts.

When we discuss the performance of such intermediate nodes, rather than the end-hosts, we have to evaluate the performance, when we have mutiple ATM connections with and without FEC-SSCS. The performance of FEC-SSCS handling with pure software processing, as shown above, will not be sufficient for an intermediate node. In order to apply the FEC-SSCS scheme in the intermediate router, we should apply the hardware processing of FEC-SSCS process in the intermediate node, in order to have a sufficient performance. The performence, when multiple ATM connections with FEC-SSCS will be handled, should be for a further study item.

Also, the discussion and evaluation performed in this paper would be worth to discuss in the case where the FEC method is applied in layer 3 (i.e., IP layer), as well as applied in ATM layer (i.e., layer 2). The discussion and the evaluation, when we apply the FEC method in layer 3 (i.e., IP), should be a for further study item.

5. Conclusion

In this paper, a novel AAL-level FEC scheme, i.e., FEC-SSCS, has been proposed and evaluated. The proposed scheme can achieve better performance than the performance with existing AAL 5, and can solve the issues in the other AAL-level FEC schemes. In the proposed scheme, the variable length user data can be efficiently transmitted with optimal redundant FEC data length according to the network condition. And the data transmission latency can be minimized through the pipelining technique.

The performance of the proposed method has been numerically evaluated, through the computer simulation, with the correlated cell loss process (i.e., cells are likely dropped consecutively), and also has been evaluated using the prototype implementation. Even with this burst cell loss, the proposed FEC scheme can achieve good performance, compared with the performance without FEC scheme. Alsom the prototype system, that perform the proposed FEC mechanism with software processing, can achieve few Mbps end-to-end throughput.

Future study items should be the following points,

.

1. The evaluated parameters (i.e., average latency and efficiency) in this paper may be less important parameters for many applications. The evaluation in terms of packet delay jitter and throughput are the likely evaluation parameters for the future study.

2. Fixed length packets are used for the evaluation in this paper, though the packet size is variable in the actual data communication. The proposed FEC scheme can autonomously modify the appended redundant data and transmit the variable user data efficiently. The impact and performance of this capability should be evaluated.

3. The commonly available control policy (i.e., flow control and error recovery policy) at upper layer is different from the model assumed in this paper. The performance with a commonly available con-

trol policy (e.g., dynamic window control with Go-back-N retransmission policy) should be evaluated.

4. Though the proposed FEC mechsnism in this paper is only for ATM networks, the proposed FEC mechsnism can be also applied to the IP packet level. When we apply the FEC mechanism to IP packet level, the FEC mechanism can apply to all data-link platforms.

Acknowledgment

This paper comes from the discussions by the e-mail group that were created after the ATM Forum February meeting in 1995. Especially, the authors thank to Dr. Georg Carle (GMD-Fokus in Germany) and Dr.Aloke Guha (in USA)

References

[1] H. Esaki, K. Kanai and Y. Tsuda, "Necessity of Cell level FEC policy for data transmission service", ATM-Forum Technical Contributions/95-150.

[2] A. Guha, H. Esaki, G. Carle and T. Dwight, "Necessity of Cell-Level FEC Scheme for ATM Networks", ATM-Forum Technical Contributions/95-1011.

[3] G. Carle, T. Dwight, H. Esaki, A. Guha, K. Tsunoda and K. Kanai, "Specification of an FEC Service Specific Convergence Sublayer for AAL5 and AAL 3/4", ATM-Forum Technical Contributions/95-326R1.

[4] K.Kanai, R.Grueter, K.Tsunoda, T.Saito, H.Esaki, "Forward Error Correction Control on AAL5 : FEC-SSCS", ICC'96, pp.384-391, June 1996.

[5] H. Esaki, Takeo Fukuda : "Reliable IP Multicast Communication Over ATM Networks Using Forward Error Correction Policy", IEICE Transactions on Communications, Vol.E78-B, No.11, November, 1995.

[6] ITU-T, Recommendation I.363 - B-ISDN ATM Adaptation Layer (AAL) Specification.

[7] R. Atkinson, "Default IP MTU for use over ATM AAL5", RFC1626, May, 1994.

[8] Biersack, E. W., "Performance Evaluation of Forward Error Correction in an ATM Environment", IEEE Journal on Selected Areas in Communication, vol. 11, Number 4, pp.631-640, May 1993.

[9] H. Ohta and T. Itami, "A Cell Loss Recovery Method Using FEC in ATM Networks", IEEE JSAC, vol. 9, pp. 1471-1483, Dec. 1991.

[10] A. J. McAuley, " Reliable Broadband Communication Using a Burst Erasure Correcting Code", ACM, pp.297-306, 1990.

Kumiko Kanai received the B.S. degree in Computer Science and the M.S.degree in Computer Science in 1990 and 1992, respectively from Gunma University. In 1992 she joined Research and Development Center, Toshiba Corporation where she has been working on communication systems. Her reseach interests are in the fields of reliable communication systems in the Internet.

Keiji Tsunoda received the B.E. and M.E. degrees from Osaka University, Japan, in 1987 and 1989, respectively. Since 1989 he has been with Research and Development Center, Toshiba Corporation as a research engineer. His current interests include applications of error control schemes, wireless and mobile systems, and broadband network technology.

Shigeo Matsuzawa received the B.E. degree from Waseda University, Japan, in 1990. He joined Research and Develelopment Center, Toshiba Corporation where he has been working on communication system. He is currently interested in internet technology for high speed networks.

Hiroshi Esaki received the B.E. and M.E. degrees from Kyushu University, Fukuoka, Japan, in 1985 and 1987, respectively. And, he received Ph.D from University of Tokyo, Japan, in 1998. In 1987, he joined Research and Development Center, Toshiba Corporeation, where he engaged in the research of ATM systems. He has been at Bellcore in New Jersey (USA) as a residential researcher from 1990 to 1991, and has engaged in the research on high speed computer communications. From 1994, he has been at CTR (Center for Telecommunications Reserch) of Columbia University in New York (USA) as a visiting scholar. He is currently interested in a high speed internet architeture and a mobile computing.

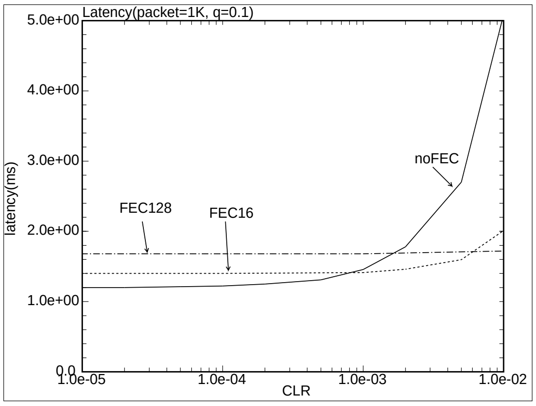

Figure 4. Latency(packet size=1K, $q = 0.1$)

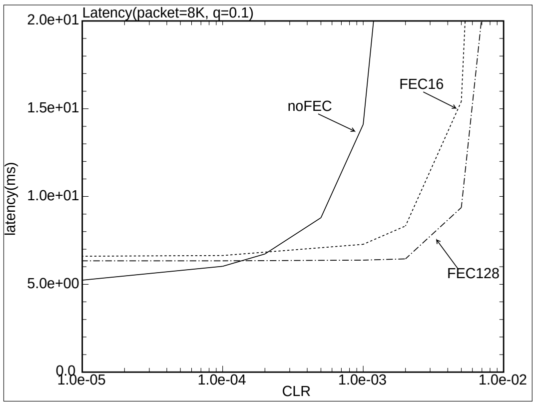

Figure 5. Latency(packet size=8K, $q = 0.1$)

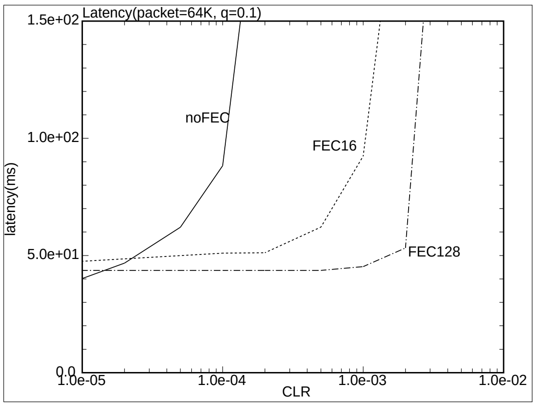

Figure 6. Latency(packet size=64K, $q = 0.1$)

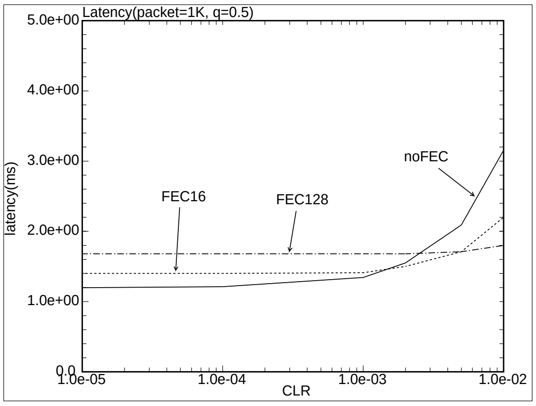

Figure 7. Latency(packet size=1K, $q = 0.5$)

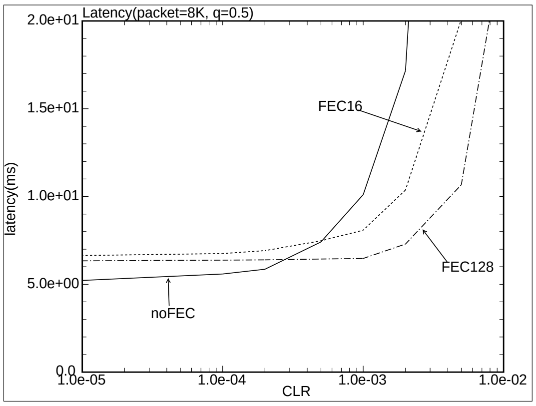

Figure 8. Latency(packet size=8K, $q = 0.5$)

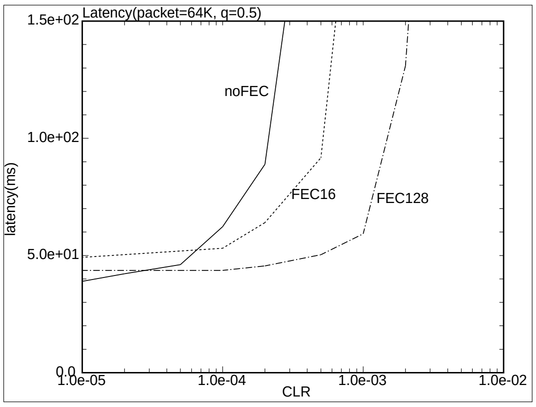

Figure 9. Latency(packet size=64K, $q = 0.5$)

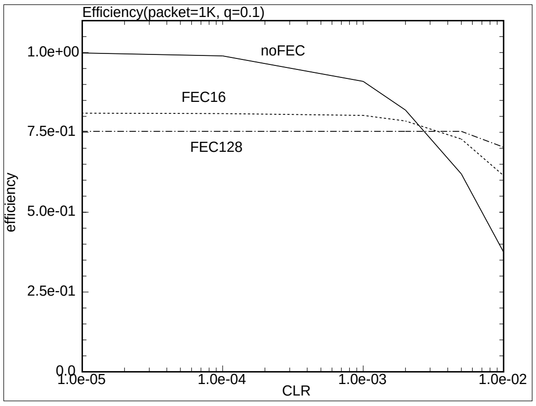

Figure 10. Efficiency(packet size=1K, $q = 0.1$)

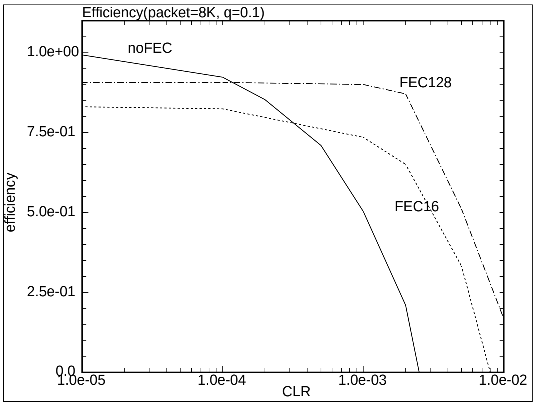

Figure 11. Efficiency(packet size=8K, $q = 0.1$)

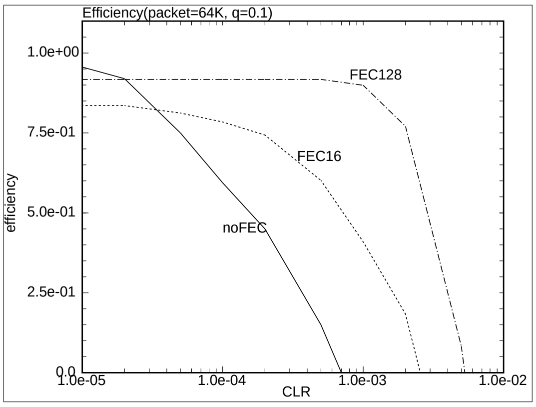

Figure 12. Efficiency(packet size=64K, $q = 0.1$)

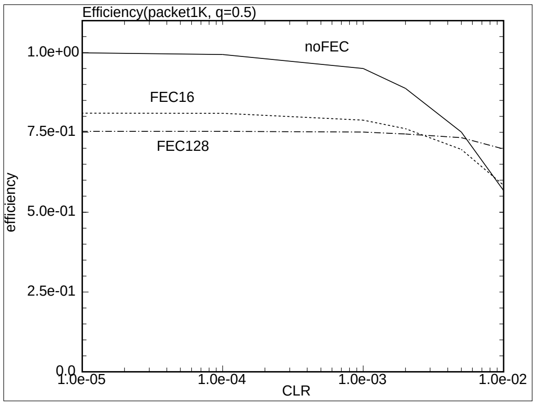

Figure 13. Efficiency(packet size=1K, $q = 0.5$)

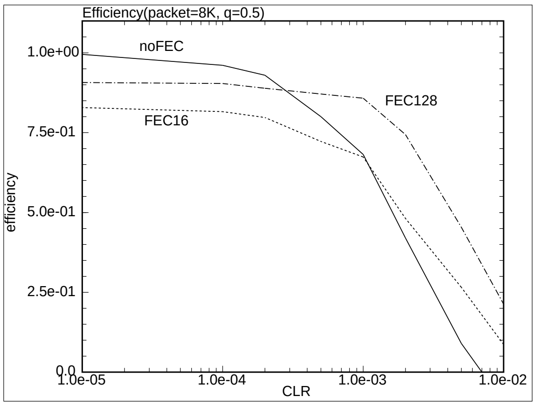

Figure 14. Efficiency(packet size=8K, $q = 0.5$)

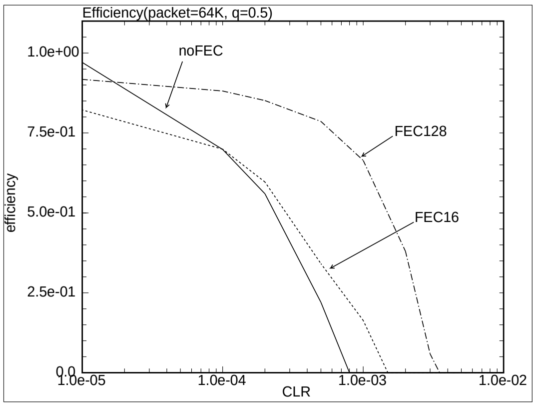

Figure 15. Efficiency(packet size=64K, $q = 0.5$)