Parity Streams: A Novel FEC Scheme with the Stream Control Transmission Protocol

Dihong Tian, *Student Member, IEEE*, and Ghassan AlRegib, *Member, IEEE*

***Abstract*— Conventional packet-level forward error correction (FEC) is found inapplicable with the stream control transmission protocol (SCTP), because of its partially ordered transport feature. To provide error resilience for respectively ordered data streams, we propose to generate "parity streams" parallel to the data streams. The use of parity streams allows multiple data streams to be protected concurrently and also scalably. For instance, unequal error protection (UEP) can be easily implemented within and/or across data streams, which is attractive to streaming-media applications where preferential treatment on different data units is desired.**

***Index Terms*— SCTP, multi-streaming, error resilience, FEC, unequal error protection, multimedia application.**

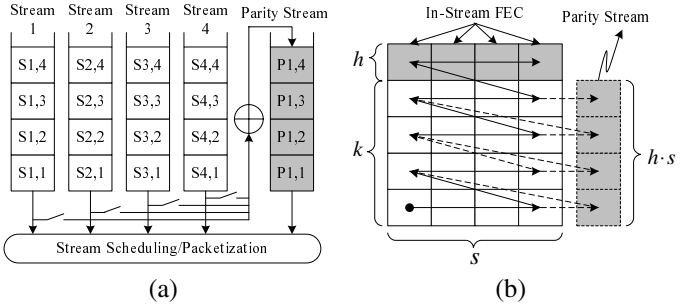


Fig. 1. Illustration on using parity streams with multi-streaming.

I. Introduction

THE STREAM control transmission protocol (SCTP) [1], [2] allows independent data streams from the upper-layer application to be multiplexed into one association and maintains respectively the order of data units on different streams, while it retains TCP-friendly congestion control and congestion avoidance. This partially-ordered transport feature, referred to as *multi-streaming*, is supported by regulating transmission at a unit of *chunk*. Each chunk is assigned a unique transmission sequence number (TSN) while chunks belonging to different streams are identified by stream identification numbers (SIDs) and stream sequence number (SSNs). Multiple chunks may be bundled into one SCTP packet for IP-transport. However, unlike TCP, which allocates sequence numbers at the packet level, there is no sequence number for SCTP packets.

To match the multi-streaming feature in SCTP, traditional packet-level FEC [3], [4] should be adjusted accordingly to present chunk-level FEC. Two implementations of chunk-level FEC are investigated in this letter. One is to append parity chunks to data chunks in different streams, respectively. Such "in-stream" FEC, although relatively easy to implement, has deficiency as one parity chunk can only provide error resilience for data chunks in the same stream whereas it introduces transmission redundancy to all the streams. In this letter, we propose to use "parity streams" to overcome the deficiency. We show that by treating parity data as "independent" streams, data streams can be protected concurrently, which accomplishes more efficient loss recovery, and scalably, which allows different data chunks to be protected unequally according to their application-related importance.

Manuscript received January 18, 2005. The associate editor coordinating the review of this letter and approving it for publication was Prof. Homayoun Yousefi'zadeh.

The authors are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA (email: {dhtian, gregib}@ece.gatech.edu).

Digital Object Identifier 10.1109/LCOMM.2006.06013.

A detailed discussion on the two chunk-level FEC schemes is presented in Section II. In Section III, we derive residual error probabilities for the two schemes on a simplified Gilbert channel and provide a numerical comparison with the derived results. We conclude the letter in Section IV.

II. Parity Streams

Let us consider common FEC codes such as Reed-Solomon codes, where for every block of k data chunks $h = (n - k)$ parity chunks are generated to form an n-chunk FEC block. Such an (n, k) systematic code can recover up to h chunk losses (erasures) as positions of the lost chunks in the chunk sequence are known.

Figure 1 illustrates our basic idea of using "parity streams" along with multi-streaming. As shown in Figure 1(a), chunks from four data streams are separately ordered for delivery. To form block erasure codes, instead of appending parity chunks to four data streams respectively, one parity stream is generated using data chunks from all the data streams. Upon transmission, this parity stream is assigned a separate stream identification number (SID). Since both data and parity chunks are uniquely indexed, the receiving end can recognize locations of lost chunks and recover the code block under the code's correcting capability. Meanwhile, partially ordered transport can be performed for all the streams, which provides more efficient use of the receiver's streaming buffer [2].

In both in-stream FEC and parity-stream cases, side information needs to be transmitted for the receiver to decode the FEC codes correctly. Efficient coding of such side information is not within the scope of this letter. We reasonably assume that overhead for sending the side information is negligible compared to the entire bitstream.

A. Equal Error Protection

Figure 1(b) shows a scenario where equal error protection is considered for s data streams. In-stream FEC appends h

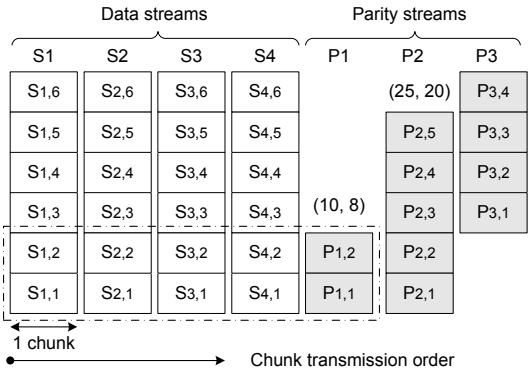


Fig. 2. Unequal error protection with parity streams.

parity chunks for every block of k data chunks in each stream, respectively, while a parity stream generates hs parity chunks for the total number of ks data chunks concurrently[1]. The solid and dashed arrows indicate chunk transmission orders in the two cases, respectively, following a round-robin scheme. In both cases, the entire FEC block under consideration has a total number of ns chunks. Let $Q_I(s)$, $Q_P(s)$ denote the probabilities that the entire block is correctly decoded in the in-stream FEC and the parity-stream cases, respectively. Then $Q_I(s) = \Pr\{\leq h$ losses out of n chunks in each of s streams$\}$, and $Q_P(s) = \Pr\{\leq hs$ losses out of ns chunks$\}$.

Proposition 1: Independent from the random loss process in the ns-chunk block, $Q_I(s) \leq Q_P(s)$ for $s \geq 1$.

The proof of Proposition 1 is trivial and is skipped here for the limited space.

B. Unequal Error Protection

Although a larger FEC block in general results in higher error-correcting ability, different data streams, as well as different data chunks in each stream, may have unequal decoding importance depending on the application and desire preferential error protection to achieve optimized rate-distortion performance. For example, an MPEG-4 synthetic audio-visual scene may consist of various audio-visual objects, each of which is independently coded into scalable resolutions. In such a scenario, various objects can be transmitted over multiple data streams regarding their coding independencies, while unequal error resilience is added to different layers of different objects through a plurality of FEC codes. With in-stream FEC, this can only be achieved by appending certain numbers of parity chunks to different data streams, while with parity streams, unequal error protection can be implemented by generating multiple parity streams for arbitrary sets of data chunks. As an example, Figure 2 shows that three parity streams are generated for four data streams, where chunks $(S_{i,j}, P_{1,j})$, $i, j = 1, 2$, form a $(10, 8)$ FEC code, and $(S_{i,j}, P_{2,j})$, $i, j = 1, 2, ..., 5$ form a $(25, 20)$ code.

It is easy to show that using parity streams for unequal error protection achieves equal or higher rate-distortion performance compared to in-stream FEC, as any UEP schemes implemented by in-stream FEC can be generated by parity streams while the reverse is not true.

[1]It should be mentioned that using a larger FEC codeword will require increased coding complexity, which may be upper-bounded by the computation resource in a particular implementation.

III. Residual Error Probabilities

Our preceding discussion presented loose comparisons between the two FEC schemes. More analytically, in this section we derive residual error probabilities of the two FEC schemes and perform numerical comparisons with derived results. We refer to the equal error protection case described in Section II-A for simplicity. Nonetheless, derivation can be easily applied to unequal error protection with minor modifications. These results are useful for devising rate-distortion optimal error-resilient schemes for multimedia applications.

We consider a simplified Gilbert channel, which is a two-state Markov model with transition probability matrix $\mathbf{P} = \begin{pmatrix} \beta & 1-\beta \\ 1-\alpha & \alpha \end{pmatrix}$ where α and β are the probabilities that chunk reception remains in failure and success states, respectively. The mean chunk loss rate p, the average burst-loss length r, and the correlation of two consecutive losses ρ are

$$p = \frac{1-\beta}{1-\alpha+1-\beta}, r = \frac{1}{1-\alpha}, \text{ and } \rho = \alpha + \beta - 1. \quad (1)$$

Let $P(\nu, m)$ be the probability of ν losses in a sequence of m chunks, $P_G(\nu, m)$ be the probability of ν losses in m chunks ending with a successfully received chunk, and $P_B(\nu, m)$ be the probability of ν losses in m chunks ending with a lost chunk. Following the literature [5], we have

$$P(\nu, m) = P_G(\nu, m) + P_B(\nu, m). \quad (2)$$

For $m = 1, 2, 3, ...$ and $\nu = 0, 1, 2, ..., m$,

$$P_G(\nu, m) = \beta P_G(\nu, m-1) + (1-\alpha) P_B(\nu, m-1), \quad (3)$$

$$P_B(\nu, m) = \alpha P_B(\nu-1, m-1) + (1-\beta) P_G(\nu-1, m-1). \quad (4)$$

The initial conditions for the recursion are

$$P_G(0,0) = \frac{1-\alpha}{1-\alpha+1-\beta}, P_B(0,0) = \frac{1-\beta}{1-\alpha+1-\beta}, \quad (5)$$

and $P_G(\nu, 0) = P_B(\nu, 0) = P_B(0, m) = 0$ for $\nu \neq 0$, $m \neq 0$.

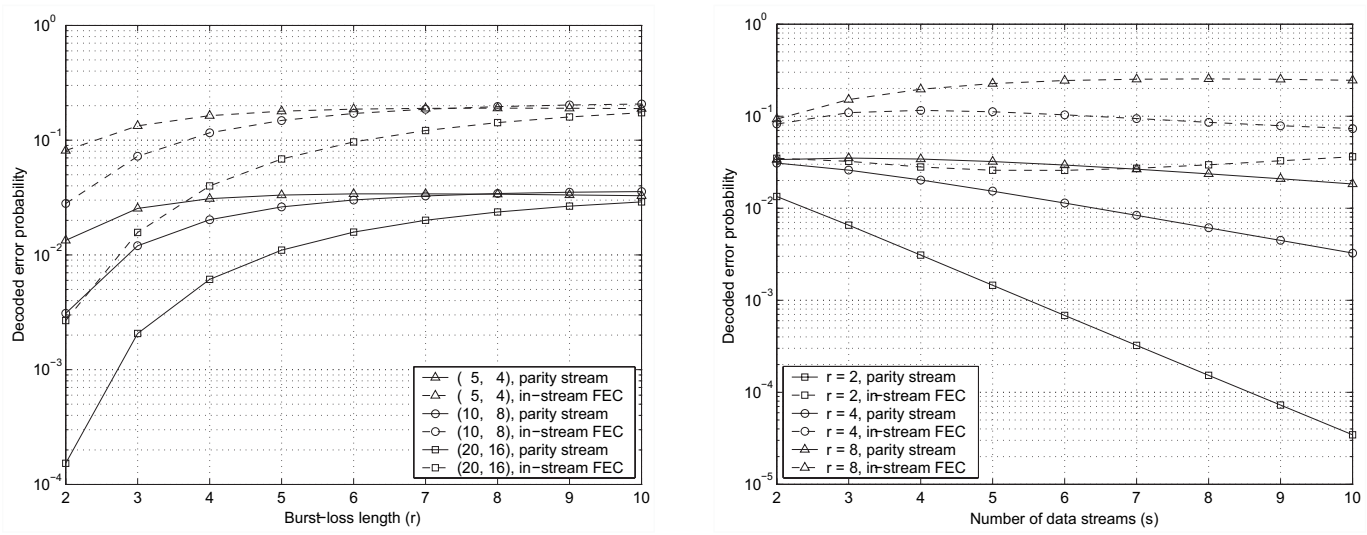
In the parity-stream case, since all the ns chunks form one FEC block, the recovering probability can be calculated by

$$Q_P(s) = \sum_{\nu=0}^{hs} P(\nu, ns). \quad (6)$$

The calculation of $Q_I(s)$, i.e., the recovering probability of the ns-chunk block in the case of in-stream FEC, is a bit complex as multiple streams are transmitted interleavingly (Figure 1(b)) and chunk losses among the interleaved streams are inter-dependent. To account for the interleaving degree, it is necessary to know transition probabilities for two chunks that are spaced apart by an arbitrary d-chunk time in transmission, which are shown in [5] to be

$$\alpha_d = p + \rho^d (1-p), \text{ and } \beta_d = (1-p) + \rho^d p. \quad (7)$$

Now, let $\widetilde{Q}_I(s; \kappa_1, \kappa_2, ..., \kappa_l)$ be the probability that streams $\kappa_1, \kappa_2, ..., \kappa_l$ are correctly decoded among all s streams, and $\widetilde{Q}_I(s; \kappa_i | \kappa_1, \kappa_2, ..., \kappa_l)$ be the conditional probability that



(a) $s = 4, p = 0.02$ (b) $(n, k) = (10, 8), p = 0.02$

Fig. 3. Numerical comparisons of decoded error probabilities between in-stream FEC and parity streams in the burst-loss case.

stream κ_i is recoverable given streams $\kappa_1, \kappa_2, ..., \kappa_l$ are all recoverable. Herein we assume $\kappa_1 < \kappa_2 < \cdots < \kappa_l$ without loss of generality. With the renewal property of a Markov process, it is easy to show that the following equality holds:

$$\widetilde{Q}_I(s; \kappa_i | \kappa_1, \kappa_2, ..., \kappa_l) = \widetilde{Q}_I(s; \kappa_i | \kappa_1, \kappa_l), \ \kappa_l < \kappa_i \leq s. \tag{8}$$

Applying (8), the probability $Q_I(s)$ can be expressed as

$$\begin{aligned} Q_I(s) &= \widetilde{Q}_I(s;1) \prod_{\kappa=2}^{s} \widetilde{Q}_I(s; \kappa | 1, \kappa - 1) \\ &= \widetilde{Q}_I(s;1) \prod_{\kappa=2}^{s} \widetilde{Q}_I(s; 1, \kappa - 1) \widetilde{Q}_I(s; 1, \kappa - 1, \kappa). \end{aligned} \tag{9}$$

To evaluate (9) we need to calculate three types of probabilities: $\widetilde{Q}_I(s;1)$, $\widetilde{Q}_I(s; 1, \kappa - 1)$ and $\widetilde{Q}_I(s; 1, \kappa - 1, \kappa)$ for $3 \leq \kappa \leq s$. $\widetilde{Q}_I(s;1)$ can be calculated using (2)–(6) by replacing the transition probabilities with α_s and β_s, as given in (7), and setting $s = 1$ in (6). Similarly, the latter two quantities can be evaluated if *joint-loss* probabilities of multiple streams are known. Generally denoting such probabilities by $\widetilde{P}_s(\nu_{\kappa_1}, \nu_{\kappa_2}, ..., \nu_{\kappa_l}, n)$ where ν_{κ_j}, $j = 1, ..., l$, is the number of losses in stream κ_j and n is the total number of chunks in each stream, we can rewrite $Q_I(s)$ as

$$\begin{aligned} Q_I(s) = \sum_{\nu_1=0}^{h} \widetilde{P}_s(\nu_1, n) \prod_{\kappa=2}^{s} \Bigg[& \sum_{\nu_1=0}^{h} \sum_{\nu_{\kappa-1}=0}^{h} \widetilde{P}_s(\nu_1, \nu_{\kappa-1}, n) \\ & \times \sum_{\nu_1=0}^{h} \sum_{\nu_{\kappa-1}=0}^{h} \sum_{\nu_\kappa=0}^{h} \widetilde{P}_s(\nu_1, \nu_{\kappa-1}, \nu_\kappa, n) \Bigg]. \end{aligned} \tag{10}$$

$\widetilde{P}_s(\nu_1, n)$ in (10) is given by (2)–(5) with properly replaced transition probabilities. Following a similar logic as (2)–(5), and using the results in (7), one can also derive recursive expressions for $\widetilde{P}_s(\nu_1, \nu_{\kappa-1}, n)$ and $\widetilde{P}_s(\nu_1, \nu_{\kappa-1}, \nu_\kappa, n)$ by listing all possible ending states of the last chunks in the considered streams. Explicit expressions are skipped here because of the limited space.

Using (6) and (10), we perform numerical comparisons between $Q_I(s)$ and $Q_P(s)$ and present the results in Figure 3, where decoded error probabilities, $1 - Q_z(s)$ with $z = P$ or I, are plotted for different parameters. In particular, Figure 3(a) shows the variation of decoded error probability with respect to the average burst-loss length, while Figure 3(b) shows the variation of decoded error probability with respect to the number of data streams. Interestingly, it is observed that $Q_P(s)$ and $Q_I(s)$ may exhibit non-monotonicity upon different selection of parameters, particularly when the average burst-loss length increases or the FEC code has lower error-correcting ability. Nevertheless, under same conditions, using parity streams always achieves much lower error probabilities than in-stream FEC, confirming the efficacy of parity streams in providing error resilience for multiple data streams.

IV. Conclusions

We proposed "parity streams" as a novel FEC scheme with the stream control transmission protocol (SCTP), innovated by the multi-streaming feature of SCTP. Analytical and numerical results showed that using parity streams achieves concurrent, and therefore more efficient, error protection than in-stream FEC. The incorporation of parity streams with multi-streaming is envisioned to facilitate the design of scalable and rate-distortion optimal streaming-media applications.

References

[1] R. Stewart, Q. Xie, K. Morneault, C. Sharp, and H. Schwarzbauer *et al.*, "Stream control transmission protocol, Request for Comments (RFC) 2960, The Internet Society, Oct. 2000.

[2] S.-J. Fu and M. Atiquzzaman, "SCTP: state of the art in research, products, and technical challenges," *IEEE Commun. Mag.*, vol. 42, pp. 64–76, Apr. 2004.

[3] C. Huitema, "The case for packet level FEC," in *Proc. IFIP Protocols for High-Speed Networks 1996*, pp. 389–300.

[4] J. Nonnenmacher, E. W. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Trans. Networking*, vol. 6, pp. 349–361, Aug. 1998.

[5] J. R. Yee and E. J. Weldon, "Evaluation of the performance of error-correcting codes on a Gilbert channel," *IEEE Trans. Commun.*, vol. 43, pp. 2316–2323, Aug. 1995.