Erschienen in: Proceedings - 2002 International Conference on Image Processing : 22 - 25 September 2002, Rochester Riverside Convention Center, Rochester, New York, USA ; vol. 2 . - Piscataway, NJ : IEEE Operations Center, 2002. - S. 165-168. - ISBN 0-7803-7622-6

PACKET LOSS PROTECTION OF EMBEDDED DATA WITH FAST LOCAL SEARCH

Vladimir Stanković, Raouf Hamzaoui

University of Konstanz
Department of Computer and Information Science
D-78457 Konstanz
stankovi,hamzaoui@fmi.uni-konstanz.de

Zixiang Xiong

Texas A &M University
College Station, TX 77843
zx@lena.tamu.edu

ABSTRACT

Unequal loss protection with systematic Reed-Solomon codes allows reliable transmission of embedded multimedia over packet erasure channels. The design of a fast algorithm with low memory requirements for the computation of an unequal loss protection solution is essential in real-time systems. Because the determination of an optimal solution is time-consuming, fast suboptimal solutions have been used. In this paper, we present a fast iterative improvement algorithm with negligible memory requirements. Experimental results for the JPEG2000, 2D, and 3D set partitioning in hierarchical trees (SPIHT) coders showed that our algorithm provided close to optimal peak signal-to-noise ratio (PSNR) performance, while its time complexity was significantly lower than that of all previously proposed algorithms.

1. INTRODUCTION

Because of the increasing demands for Internet and wireless multimedia products, such as video streaming and third-generation mobile phones, the design of robust real-time systems for the transmission of multimedia over unreliable communication channels has become a major research topic of the research community over the last years [1, 2, 3, 4].

Some of the most popular models used to describe such channels are the Rayleigh fading channel and the packet erasure channel. Once the channel model and the communication system have been selected, the main challenge is to obtain the best possible performance at a target transmission rate. However, for many real-time systems, both time complexity and memory requirements also play an essential role.

In this paper, we consider a system that sends an embedded bitstream over a packet erasure channel. Embedded bitstreams can be generated by coders like SPIHT [5] and JPEG2000 [6] for images, and 3D SPIHT [7] for video sequences. We study solutions that protect the source bitstream with Reed-Solomon codes [3, 8, 9, 10, 11]. We assume that the joint source-channel code is sent in N packets, each of which has a fixed length of L symbols.

Several researchers devised efficient unequal loss protection solutions for such systems. Mohr, Riskin, and Ladner [8] proposed a local search algorithm. In [9], they presented a faster algorithm. Given $p = LN$ points on the operational distortion-rate curve of the source coder, the algorithm first computes the h vertices of their convex hull. Then, a solution is found in $O(hN \log N)$ time. This solution is optimal under the assumption of the convexity of the distortion-rate function and of fractional bit allocation assignment. Puri and Ramchandran [3] provided a Lagrange multiplier-based algorithm. The algorithm also starts by computing the h vertices of the convex hull of p points of the operational distortion-rate curve. Then, after an $O(h)$ step, it requires several Lagrange iterations, each of which takes $O(N)$ time. Stockhammer and Buchner [10] presented an $O(N^2L^2)$ dynamic programming algorithm that is close to optimal in the general case and optimal if the operational distortion-rate function is convex and the packet loss probability is a monotonically decreasing function of the number of packets. Dumitrescu, Wu, and Wang [11] independently found the same algorithm. However, they showed that its complexity can be reduced to $O(NL^2)$. Moreover, they gave an $O(N^2L^2)$ algorithm that is optimal in the general case.

The paper is organized as follows. In Section 2, we introduce our terminology and state the optimal unequal loss protection problem as a combinatorial optimization problem. In Section 3, we present an $O(NL)$ local search algorithm that starts from a solution that maximizes the expected number of received source bits and iteratively improves this solution. The algorithm is inspired by our previous work [12], which was developed in the context of joint source-channel coding in memoryless noisy channels. In Section 4, we compare the PSNR performance and the speed of our algorithm to that of the previous ones for the 2D and 3D SPIHT source coders and for JPEG2000. Our algorithm provided similar PSNR performance to that of the state-of-the-art, while both its memory requirements and time complexity were lower.

2. PROBLEM STATEMENT

In this section, we introduce our terminology and state the packet loss protection problem as a combinatorial optimization problem. Our notation closely follows that of [11].

Suppose that an embedded source bitstream is to be protected and sent through an erasure channel as N packets of L symbols (for example, bytes) each. A popular protection system [3, 8, 9] builds L source segments $S_1, \ldots, S_L$ of $m_i \in \{1, \ldots, N\}$ symbols each and protects each segment with an (N, m_i) systematic Reed-Solomon (RS) code of maximal distance. For each $i = 1, \ldots, L$, let $f_i = N - m_i$ denote the number of RS redundancy symbols that protect segment S_i. If n packets of N are lost, then the RS codes ensure that all segments that contain at most $N - n$ source symbols can be recovered. Thus, by adding the constraint that $f_1 \geq f_2 \geq \cdots \geq f_L$, if at most f_i packets are lost, then the receiver can decode at least the first i segments. In the fol-

Konstanzer Online-Publikations-System (KOPS)
URL: http://nbn-resolving.de/urn:nbn:de:bsz:352-231289

lowing, we denote by $\mathcal{F}$ the set of L-tuples $(f_1, \ldots, f_L)$ such that $f_i \in \{0, \ldots, N-1\}$ for $i = 1, \ldots, L$ and $f_1 \geq f_2 \geq \cdots \geq f_L$. Let $p_N(n)$ denote the probability of losing exactly n packets of N and let $c_N(k) = \sum_{n=0}^{k} p_N(n),\ k = 0, \ldots, N$. Then $c_N(f_i)$ is the probability that the receiver correctly recovers segment S_i. Let ϕ denote the distortion-rate function of the source coder and let X be the random variable whose value is the number of packets lost. An optimal packet loss protection for the system consists of finding a protection scheme $F = (f_1, \ldots, f_L) \in \mathcal{F}$ that minimizes the expected distortion

$$E_L[d](f_1, \ldots, f_L) = \sum_{i=0}^{L} P_i(F)\phi(r_i), \tag{1}$$

where $P_0(F) = P(X > f_1), P_i(F) = P(f_{i+1} < X \leq f_i)$ for $i = 1, \ldots, L-1,\ P_L(F) = P(X \leq f_L), r_0 = 0$, and $r_i = \sum_{k=1}^{i} m_k = iN - \sum_{k=1}^{i} f_k$ for $i = 1, \ldots, L$. Note that for $i = 1, \ldots, L-1$, we have $P_i(F) = 0$ if $f_i = f_{i+1}$ and $P_i(F) = \sum_{n=f_{i+1}+1}^{f_i} p_N(n)$, otherwise. In [11], the expected distortion (1) was given in the equivalent form

$$c_N(N)\phi(r_0) + \sum_{i=1}^{L} c_N(f_i)(\phi(r_i) - \phi(r_{i-1})). \tag{2}$$

3. LOCAL SEARCH ALGORITHM

In this section, we propose a local search algorithm that finds an approximately optimal solution to the problem of minimizing (1). From our previous work [12], it follows that if the operational distortion-rate (respectively PSNR-rate) function of the source coder is nonincreasing (respectively nondecreasing) and convex (respectively concave), which is a reasonable assumption for a large class of embedded coders, then the total number of protection symbols $\sum_{i=1}^{L} f_i^*$ of a distortion-optimal solution $(f_1^*, \ldots, f_L^*)$ is greater than or equal to that of a rate-optimal solution. By rate-optimal solution, we mean a solution that maximizes the expected number of received source bits

$$E_L[r](f_1, \ldots, f_L) = \sum_{i=0}^{L} P_i(F) r_i. \tag{3}$$

Our local search algorithm works by iterative improvement. We start at a rate-optimal solution and search for the best candidate in its neighborhood. If this candidate is better than the current solution, we adopt it and repeat the search from the new solution. Otherwise, we stop. Note that the computation of a rate-optimal solution is straightforward. Indeed, we have

$$E_L[r](f_1, \ldots, f_L) \quad = \quad \sum_{j=1}^{L} m_j \sum_{i=0}^{f_j} p_N(i). \tag{4}$$

Thus, by setting

$$f_r = \arg \max_{i=0, \ldots, N-1} (N-i) \sum_{n=0}^{i} p_N(n),$$

one can deduce from (4) that a rate-optimal solution is the equal loss protection solution $(f_r, \ldots, f_r)$.

We now specify the neighborhood of a solution. In accordance with the result in [12], the neighborhood of a solution is restricted to solutions that provide a stronger protection.

Definition 1 *Let $F = (f_1, \ldots, f_L) \in \mathcal{F}$. The neighborhood of F consists of the solutions of the form $(f_1 + 1, f_2, \ldots, f_L), (f_1 + 1, f_2 + 1, \ldots, f_L), \ldots, (f_1 + 1, f_2 + 1, \ldots, f_{L-1} + 1, f_L + 1)$ that are included in $\mathcal{F}$.*

For example, suppose that $L = 4$ and $N = 8$. Then the neighbors of $F = (6, 6, 5, 4)$ are the solutions $(7, 6, 5, 4), (7, 7, 5, 4), (7, 7, 6, 4)$, and $(7, 7, 6, 5)$. Note that the solutions that do not belong to $\mathcal{F}$ are not considered. For example, the set of neighbors of $F = (7, 6, 5, 4)$ is empty.

In the worst case, our algorithm starts from the rate-optimal solution $(0, \ldots, 0)$ and stops at $(N-1, f_2, \ldots, f_L)$. The determination of the rate-optimal solution can always be done in $O(N)$ steps. The refinement process needs $L(N-1)+1$ computations and $L(N-1)$ comparisons of the cost function (1), respectively. When we compute the cost function for the neighbors of a solution, we exploit the fact that two successive neighbors differ in only one segment. Thus, only two probabilities P_i have to be recomputed. For example, let $F_1 = (7, 6, 5, 4)$ and $F_2 = (7, 7, 5, 4)$. Then $P_0(F_2) = P_0(F_1), P_3(F_2) = P_3(F_1)$, and $P_4(F_2) = P_4(F_1)$. The overall worst-case complexity of our algorithm is thus $O(NL)$.

Because our algorithm exploits the assumption of the convexity (respectively concavity) of the operational distortion-rate (respectively PSNR-rate) function of the source coder, it may provide unsatisfactory results when the convexity assumption is severely violated. For example, we obtained much better results for the SPIHT coder than for JPEG2000 (see Figure 1 for a comparison of the corresponding PSNR-rate curves). To overcome this problem, we propose for JPEG2000 to compute our solution by applying the local search algorithm to the piecewise affine function obtained by joining the points of the operational PSNR-rate curve at which the PSNR changes. An alternative would be to use the upper convex hull of the operational PSNR-rate curve.

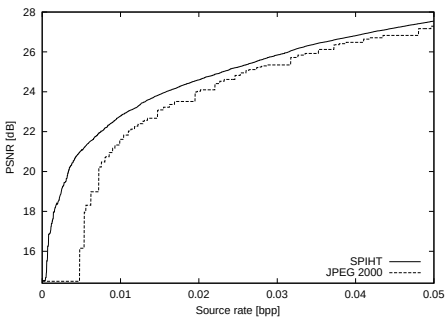

Fig. 1. Operational PSNR-rate functions of the SPIHT and JPEG2000 coders for the 512×512 Lenna image.

4. EXPERIMENTAL RESULTS

We compared the time complexity and the PSNR performance of our local search algorithm to that of the following algorithms:

- The local search algorithm of Mohr et al. [8]. The quality of the solution is dependent on a search parameter Q that has to be fixed ahead of time. However, there is a trade-off between quality and computing time. Since time is a major issue in our comparison, we used the algorithm with $Q = 1$, which generally yields the fastest results.

- The convex hull-based algorithm of Mohr et al. [9].
- The algorithm of Puri and Ramchandran [3].
- The two algorithms of Dumitrescu et al. [11]. Note that one of these algorithm finds an optimal solution.

The goal was to maximize the expected PSNR. Thus, the objective function was $\sum_{i=0}^{L} P_i(F)PSNR(r_i)$, where $PSNR(r_i)$ is the PSNR corresponding to source rate r_i. In our first experiments, we had $L = 48$ bytes, and the packet loss probability function $p_N(n)$ was exponentially decreasing in n with a mean loss rate of 0.2. All programs were run on a PC with a Windows operating system having an AMD Athlon (TM) XP 1600+ 1410 MHz processor with a main memory size of 1 Gbyte. Table 1 and Table 2 show the PSNR in dB and the time in seconds vs. the number of packets N for, respectively, the SPIHT and JPEG2000 bitstreams of the 8 bits per pixel (bpp) gray-scale 512 $\times$ 512 Lenna image. The JPEG2000 bitstream was generated with the Kakadu C++ implementation of [6] with the default settings. Table 3 shows the results for the 3D SPIHT bitstream of the Y component of the 176 $\times$ 144 Foreman video sequence.

For JPEG2000, we computed our solution by applying the local search algorithm to the piecewise affine function associated to the PSNR-rate curve (see Section 3). In the tables, however, we give the expected PSNR of our solution for the original PSNR-rate curve.

The CPU times take into account all operations required by an algorithm. This includes, for example, computing the convex hull in algorithms [3, 9] and computing the piecewise affine function for JPEG2000 in our algorithm. However, for all algorithms, we did not include the preprocessing time used to store the file containing the operational PSNR-rate curve in an array.

Because the optimal algorithm of [11] needs to store N^2L^2 floating point numbers and N^2L^2 integers, our computer ran out of memory for many values of N, and we were not able to report the results. The same limitation was observed with their second algorithm, which needs the storage of N^2L floating point numbers and N^2L integers.

The tables show that our local search algorithm was always the fastest algorithm. For the SPIHT bistream, the speed-up factor over the algorithm of [3] was about 10, while it was between four and six for the JPEG2000 and 3D SPIHT bitstreams. Compared to the algorithm of [9], the speed-up factor increased monotonically with N, reaching 50 for the SPIHT coder with $N = 1000$. On the other hand, the PSNR performance of our solution was similar to that of these two algorithms. This performance was close to optimal (the observation was possible only for the values of N for which it was possible to compute an optimal solution). The algorithm of [8] was much slower, and it yielded a poor solution for many large values of N. When this algorithm was used with the search parameter Q set to N (this setting gives the highest PSNR), it produced a slightly better expected PSNR than our algorithm, but the gain did not exceed 0.08 dB, and its computing time was up to 1000 times higher than that of our algorithm. The optimal algorithm of [11] was the slowest. Its CPU time was not in accordance with the theory. This is due to its huge memory requirements.

Table 4 presents results for the 3D SPIHT bitstream of the Foreman sequence when $L = 1000$ bytes and the packet mean loss rate was 0.05. Here the CPU time is given for a 270 MHz MIPS R12000 processor of an SGI Origin200 with a main memory size of 1536 Mbytes. The table shows that for large L and small N, our algorithm was slower than the algorithms of [3] and [9]. This indicates that the time complexity of our algorithm is more sensitive to increasing L than the time complexity of the algorithms in [3] and [9]. Note, however, that our algorithm was again faster when enough packets were sent.

5. CONCLUSION

We proposed an iterative improvement algorithm for RS-based unequal loss protection of embedded data in erasure channels. Our algorithm provides state-of-the-art PSNR performance with lower complexity than that of the best previous solutions.

Acknowledgments. We are very grateful to Sorina Dumitrescu, Xiaolin Wu, and Zhe Wang for providing us with the C-codes of their algorithms [11]. Many thanks to Daniel Sachs and Kannan Ramchandran for sending us the C-code used in [3]. We also thank Alexander Mohr, Richard Ladner, and Eve Riskin for making the C-code of algorithm [9] available.

6. REFERENCES

[1] Davis, G., Danskin, J., *Joint source and channel coding for image transmission over lossy packet networks, SPIE Conf. Wavelet Applications of Digital Image Proc. XIX*, pp. 376–387, Denver, August 1996.

[2] Sherwood, P. G., Zeger, K., *Error protection for progressive image transmission over memoryless and fading channels*, IEEE Trans. on Communications 46,12 (1998) 1555–1559.

[3] Puri, R., Ramchandran, K., *Multiple description coding using forward error correction codes*, in: *Proc. 33rd Asilomar Conf. on Signals and Systems*, Pacific Grove, CA, Oct. 1999.

[4] Sachs, D. G., Anand, R., Ramchandran, K., *Wireless image transmission using multiple-description based concatenated code, Proc. SPIE'00*, vol. 3974, pp. 300–311, Jan. 2000.

[5] Said, A., Pearlman, W. A., *A new fast and efficient image codec based on set partitioning in hierarchical trees, IEEE Trans. Circuits and Systems for Video Technology*, vol. 6, pp. 243–250, June 1996.

[6] Taubman, D., Marcellin, M., *JPEG2000: Image Compression Fundamentals, Standards, and Practice*, Kluwer, 2002.

[7] Kim, B.-J., Xiong, Z., Pearlman, W. A., *Low bit-rate scalable video coding with 3D set partitioning in hierarchical trees (3D SPIHT)*, IEEE Trans. Circuits and Systems for Video Tech. vol. 10, pp. 1365–1374, Dec. 2000.

[8] Mohr, A. E., Riskin, E. A., Ladner, R. E., *Unequal loss protection: graceful degradation of image quality over packet erasure channels through forward error correction*, IEEE Journal on Selected Areas in Comm. 18,7 (2000) 819–828.

[9] Mohr, A., Ladner, R., Riskin, E., *Approximately optimal assignment for unequal loss protection, Proc. IEEE ICIP-2000*, vol. 1, pp. 367–370, Vancouver, Sept. 2000.

[10] Stockhammer, T., Buchner, C., *Progressive texture video streaming for lossy packet networks, Proc. 11th International Packet Video Workshop*, Kyongju, May 2001.

[11] Dumitrescu, S., Wu, X., Wang, Z., *Globally optimal uneven error-protected packetization of scalable code streams, Proc. DCC'02*, Snowbird, Utah, April 2002.

[12] Hamzaoui, R., Stanković, V., Xiong, Z., *Rate-based versus distortion-based optimal joint source-channel coding, Proc. DCC'02*, pp. 67–72, Snowbird, Utah, April 2002.

	Da [11]		Db [11]		Ma [8]		Mb [9]		P [3]		LS	
N	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
100	28.27	1.51	28.27	0.23	28.26	0.33	28.26	0.05	28.24	0.03	28.26	<0.01
200	30.99	8.85	30.99	0.58	30.95	1.49	30.97	0.10	30.95	0.06	30.93	0.01
300	32.65	35.96	32.65	0.89	32.45	1.39	32.64	0.20	32.62	0.10	32.60	0.01
400			33.81	1.10	33.84	4.29	33.87	0.33	33.88	0.15	33.84	0.02
600			34.57	1.81	35.37	8.12	35.6	0.72	35.57	0.24	35.58	0.02
800					36.65	13.36	36.81	1.33	36.79	0.32	36.79	0.03
1000					37.18	14.20	37.85	2.15	37.81	0.41	37.84	0.04

Table 1. CPU time in seconds and expected PSNR in dB for the SPIHT bitstream of the 512×512 Lenna image. The results are given for N packets of $L = 48$ bytes each. The packet mean loss rate of the erasure channel is 0.2. Da denotes the optimal algorithm of [11], Db denotes the second algorithm of [11], Ma denotes the algorithm of [8], Mb denotes the algorithm of [9], P denotes the algorithm of [3], and LS is our local search algorithm.

	Da [11]		Db [11]		Ma [8]		Mb [9]		P [3]		LS	
N	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
100	28.00	1.50	27.72	0.25	27.13	0.12	27.98	0.05	27.80	0.02	27.94	<0.01
200	30.80	8.78	30.77	0.54	27.77	0.21	30.79	0.09	30.74	0.04	30.76	0.01
300	32.58	34.13	32.57	0.83	30.86	0.93	32.57	0.15	32.52	0.06	32.50	0.01
400			33.75	1.13	30.52	0.86	33.79	0.22	33.76	0.10	33.73	0.02
600			34.58	1.83	30.98	1.42	35.60	0.46	35.57	0.15	35.55	0.03
800					31.74	2.46	36.77	0.83	36.73	0.18	36.78	0.04
1000					35.38	27.78	37.73	1.34	37.70	0.20	37.69	0.05

Table 2. CPU time in seconds and expected PSNR in dB for the JPEG2000 bitstream of the 512×512 Lenna image. The results are given for N packets of $L = 48$ bytes each. The packet mean loss rate of the erasure channel is 0.2.

	Db [11]		Ma [8]		Mb [9]		P [3]		LS	
N	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
400	21.90	1.12	21.82	2.35	21.92	0.12	21.86	0.04	21.92	0.01
600	22.87	1.84	22.88	7.62	23.06	0.19	23.05	0.07	23.04	0.01
800			23.31	7.11	23.80	0.28	23.78	0.10	23.77	0.02
1000			24.37	22.28	24.39	0.37	24.38	0.13	24.34	0.03
1200			24.75	37.61	24.94	0.57	24.92	0.17	24.92	0.04
1400			24.84	31.82	25.40	0.78	25.36	0.24	25.36	0.05
1600			25.04	28.33	25.80	1.04	25.76	0.27	25.73	0.05
1800			25.43	33.24	26.07	1.31	26.05	0.33	26.01	0.06

Table 3. CPU time in seconds and expected PSNR in dB for the 3D SPIHT bitstream of the Foreman sequence. The results are given for N packets of $L = 48$ bytes each. The packet mean loss rate of the erasure channel is 0.2.

	Mb [9]		P [3]		LS	
N	PSNR	Time	PSNR	Time	PSNR	Time
500	35.36	1.30	35.34	1.95	35.34	2.13
1000	39.76	3.36	39.77	6.12	39.73	3.95
1500	43.01	6.52	43.02	10.21	43.05	5.65
2000	45.88	10.29	45.88	13.76	45.81	6.74

Table 4. CPU time in seconds and expected PSNR in dB for the 3D SPIHT bitstream of the Foreman sequence. The results are given for N packets of $L = 1000$ bytes each. The packet mean loss rate of the erasure channel is 0.05.