

# OPTIMAL PROTECTION ASSIGNMENT FOR SCALABLE COMPRESSED IMAGES

*Johnson Thie and David Taubman*

University of New South Wales,  
Sydney, Australia

j.thie@ee.unsw.edu.au, d.taubman@unsw.edu.au

## ABSTRACT

This paper is concerned with the efficient transmission of scalable compressed images over lossy communication channels. Recent works have proposed several strategies for assigning optimal code-rates to elements in a scalable data stream, under the assumption that all elements are encoded onto a common group of network packets. When the size of the data to be encoded becomes large in comparison with the size of the network packets, such schemes require very long channel codes with high computational complexity. In networks with high loss, small packets are generally more desirable than long packets. This paper proposes a strategy for assigning optimal code-rates to elements of the scalable compressed data source, subject to constraints on the packet size, transmission length and code complexity. Our experimental results suggest that the proposed scheme can outperform previously proposed code-rate assignment policies, subject to the above-mentioned constraints, particularly with high loss.

## 1. INTRODUCTION

In this paper we are concerned with the efficient transmission of scalable compressed images over lossy communication channels. Over the last ten years, scalable compression techniques have been widely explored, with excellent examples including the well-known EZW[1] and SPIHT [2] algorithms and, most recently, the JPEG2000[3] image compression standard. An important advantage of a scalable data stream is that part of the data may be lost by a communication network, without compromising the usefulness of the remainder of the data.

For the present work, we restrict our attention to “erasure” channels. An erasure channel is one whose data is partitioned into a sequence of elements, each of which either arrives at the destination without error or is entirely lost. The erasure channel is a good model for modern packet networks, in which the elements of interest are the network’s packets, each of which either arrives at the destination or is lost, due to congestion or corruption. A key property of the erasure channel is that the receiver knows which packets have been lost.

In the context of the erasure channel, Albanese et al.[?] pioneered an unequal error protection scheme known as Priority Encoded Transmission (PET). The PET scheme works with a family of channel codes, all having the same coded output length of  $N$  symbols, but different input lengths,  $k \leq N$ . We consider only “perfect codes,” having the property that receipt of any  $k$  out of the  $N$  symbols is sufficient to recover the  $k$  source symbols. The code-rate,  $R(k) = N/k$ , thus determines strength of the code. Given a data source consisting of elements (we will call these layers)

$\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_L$ , having uncoded lengths  $M_1, M_2, \dots, M_L$ , and code-rates,  $R(k_1) \geq R(k_2) \geq \dots \geq R(k_L)$ , the PET scheme packages the coded elements into  $N$  network packets, in such a way that receipt of any  $k$  out of the  $N$  packets is sufficient to recover all layers  $\mathcal{L}_l$  with  $k_l \leq k$ . The total length of the encoded transmission is  $\sum_l R(k_l) M_l$ , which must be arranged within  $N$  packets. As we shall see, these constraints form the motivation for a modified encoding strategy proposed in this paper.

Given the PET encoding scheme and a scalable data source, we must determine an appropriate assignment of code-rates to elements of the source. Mohr et al.[?] and Puri et al.[?] have proposed algorithms for optimizing the code-rate assignment under the following conditions. A constraint,  $M_{\max}$ , is imposed on the total coded transmission length. The optimization objective is an expected utility,  $U$ , associated with the decompressed image.  $U$  must be an additive function of the source elements which are correctly received. That is,

$$U = \sum_{l=1}^L U_l \cdot P(k_l) \quad (1)$$

where

$$P(k_l) = \sum_{k=k_l}^N \binom{N}{k} p^k (1-p)^{N-k}$$

is the probability that the number of lost packets does not exceed  $N - k_l$  and  $p$  is the individual packet recovery probability. As an example,  $-U$  might be taken to be the Mean Squared Error (MSE) of the reconstructed image and  $U_l$  represents the amount by which receipt of layer  $\mathcal{L}_l$  decreases the image MSE.

It is also useful to express  $p$  in terms of bit error probability. Suppose that  $\epsilon$  denotes the probability of one bit error and  $(1 - \epsilon)^8$  is the probability of one correct byte. Hence, the probability to recover a packet is  $p = (1 - \epsilon)^{8S}$ , where  $S$  is the packet size.

Unfortunately, these optimization strategies rely upon the PET encoding scheme, which requires all of the coded source data to be distributed across the same  $N$  packets. If the packet size is small, or the amount of coded data is large, it is necessary to work with perfect  $(N, k)$  channel codes having very large values of  $N$ . This imposes a growing computational burden on both the encoder and the decoder, but especially on the decoder. For example, suppose we wish to protect a 1 Mbyte source, using ATM cells with 45 payload bytes as the packets. The channel codes must then have lengths of  $N = 22,000$  symbols!

In this paper we present a strategy for assigning source layers from a highly scalable source to smaller “chunks” of packets, each chunk having its own set of codes with reduced lengths,  $N$ .

Given a bound on the code length,  $N$ , a packet size,  $S$ , and a maximum transmission length,  $M_{\max}$ , we find the optimal assignment of source layers to chunks and the corresponding optimal assignment of code-rates,  $R(k_i)$ , to source layers, so as to maximize the expected utility,  $U$ . We provide experimental results in the context of JPEG2000 data streams. When working with small packets and constrained code lengths, the proposed assignment strategy can significantly improve the expected utility (decrease the MSE), relative to that obtained using previously proposed code-rate assignment strategies.

## 2. SCALABLE DATA DEPENDENCIES

Scalable data is composed of nested elements. The compression of these elements generally imposes dependencies among the elements. These dependencies owe much to the way the elements are coded. In particular, elements may be coded in a way which depends upon other elements. Hence, prior to decoding any given element, other elements must first need to be successfully decoded.

An image compressed with JPEG2000 is a good example, since it can have a combination of dependent and independent elements. Dependencies exist between successive “quality layers” within the JPEG2000 data stream, where an element which contributes to a higher quality layer cannot be decoded without first decoding elements from lower quality layers. JPEG2000 also contains elements which exhibit no such dependencies. In particular, subbands from different levels in the Discrete Wavelet Transform (DWT) are coded and represented independently within the data stream. Similarly, separate colour channels within a colour image are also coded and represented independently within the data stream.

For simplicity, in this paper we restrict our attention to scalable data sources with a single linear chain of dependencies. We express such dependencies through the relationship  $\mathcal{L}_1 \prec \mathcal{L}_2 \prec \dots \prec \mathcal{L}_L$ , meaning that layer  $\mathcal{L}_1$  must be decoded before the information in layer  $\mathcal{L}_2$  can be used, and so forth. A JPEG2000 data stream with only one colour component (monochrome) and no levels of DWT has such a structure. While the algorithm can be applied to data sources with independent components, the resulting code-rate assignments will not generally be optimal. Such scenarios are the subject of other work, not documented in this brief paper.

## 3. SINGLE CHUNK ASSIGNMENT

In this section we review the problem of assigning an optimal set of channel codes (equivalently, code-rates) to the layers of a scalable data source, subject to the assumption that all source layers will be packed into the same set of  $N$  network packets, where  $N$  is the codeword length. This is the problem addressed by Mohr et al.[?] and Puri et al.[?], which we will identify as the “single chunk” assignment problem. In Section 4 we will consider the more general problem in which constraints on the codeword length force the separation of the data into multiple chunks.

As noted above, we assume a simple chain of dependencies amongst the source layers, with  $\mathcal{L}_1 \prec \mathcal{L}_2 \prec \dots \prec \mathcal{L}_L$ . We further assume that the utility-length behaviour of the source layers is convex  $\cap$ . That is,  $\frac{U_1}{M_1} \geq \frac{U_2}{M_2} \geq \dots \geq \frac{U_L}{M_L}$ . If the source is not already convex, the convex hull may be taken by aggregating layers in a suitable fashion. The elements of a JPEG2000 data stream, constructed in accordance with the usual

Post-Compression Rate-Distortion optimization strategy [4], always have convex  $\cup$  distortion-length characteristics with respect to MSE. Thus, taking  $-U$  as MSE ensures that the utility-length characteristic will be convex  $\cap$  as supposed.

If we temporarily ignore the fact that layer  $\mathcal{L}_i$  can be decoded only if layers  $\mathcal{L}_1$  through  $\mathcal{L}_{i-1}$  are successfully decoded, our task is to optimize the utility function,  $U$ , given in equation (1), subject to the overall transmission length constraint

$$\sum_{i=1}^L M_i R(k_i) \leq M_{\max}$$

This constrained optimization problem may be converted to a family of unconstrained problems, parametrized by a quantity  $\lambda > 0$ . Specifically, let  $U^{(\lambda)}$  and  $M^{(\lambda)}$  denote the expected utility and transmission length associated with the set of code-rates,  $k_i^{(\lambda)}$ , which maximize the functional

$$U^{(\lambda)} - \lambda M^{(\lambda)} = \sum_{i=1}^L \left( U_i P(k_i^{(\lambda)}) - \lambda M_i R(k_i^{(\lambda)}) \right) \quad (2)$$

Evidently, there is no way to increase  $U$  beyond  $U^{(\lambda)}$ , without also increasing  $M$  beyond  $M^{(\lambda)}$ . Thus, if we can find  $\lambda$  such that  $M^{(\lambda)} = M_{\max}$ , the  $k_i^{(\lambda)}$  will form an optimal solution to the unconstrained problem. In practice, the discrete nature of the problem may prevent us from finding a value of  $\lambda$  such that  $M^{(\lambda)} = M_{\max}$ , but if the layers are small enough, we should be justified in ignoring this small source of sub-optimality and selecting the smallest value of  $\lambda$  such that  $M^{(\lambda)} \leq M_{\max}$ .

Given  $\lambda$ , maximizing the functional in equation (2) is equivalent to maximizing  $U_i P(k_i^{(\lambda)}) - \lambda M_i R(k_i^{(\lambda)})$ , separately for each layer  $\mathcal{L}_i$ , so that the solution can be found easily by considering each of the available channel codes in turn.

Up to this point we have been ignoring the dependencies between the source elements, assuming that the utility of layer  $\mathcal{L}_i$  depends only on the probability that it is successfully received and hence the code-rate assigned to that layer alone. Fortunately, the code-rate assignments,  $k_i^{(\lambda)}$ , produced by the optimization procedure described above have the property that  $k_1^{(\lambda)} \leq k_2^{(\lambda)} \leq \dots \leq k_L^{(\lambda)}$ . This may be shown to be a consequence of the assumed convexity of the source,  $\frac{U_1}{M_1} \geq \frac{U_2}{M_2} \geq \dots \geq \frac{U_L}{M_L}$ , together with the convexity of the family of channel codes. This property means that whenever any given layer is received correctly, the lower layers on which it depends are guaranteed also to be received correctly, satisfying the required dependencies.

## 4. MULTIPLE CHUNK ASSIGNMENT

We turn our attention now to the problem of optimal code-rate assignment when source layers must be broken into separate chunks, each with their own channel codes. Since each chunk has its own set of channel codes, we no longer have the guarantee that a source layer with a stronger code will be recovered whenever a source layer with a weaker code is recovered. This was a consequence of the PET scheme which requires all layers to be coded together in a single chunk. The code-rate assignment strategy described in Section 3 relied upon this property to ensure that layer dependencies were satisfied, allowing us to use equation (1) for the expected utility.

Consider a collection of  $C$  chunks,  $\{\mathcal{C}_1, \dots, \mathcal{C}_C\}$ , characterized by parameters  $\{c_1, \dots, c_C\}$ , where  $c_m$  contains the index of the first source layer residing in chunk  $\mathcal{C}_m$ . As in the single chunk case described in Section 3, recovery of any layer,  $l \in [c_m, c_{m+1})$  within chunk  $\mathcal{C}_m$  guarantees the recovery of all previous layers in the same chunk. However, in order for these layers to be correctly decoded, all layers from all previous chunks must also be correctly recovered. Equivalently, it is sufficient for the last layer in each previous chunk to be recovered at the receiver. Since chunks are protected independently, the effective utility of layer  $\mathcal{L}_l$  which resides in chunk  $\mathcal{C}_m$  is  $U_{\text{mod}_l} = U_l \prod_{n=2}^m P(k_{c_{n-1}})$ . This changes the expected utility in equation (1) into

$$U = \sum_{m=1}^C \sum_{l=c_m}^{c_{m+1}-1} \left( U_l \left( \prod_{n=2}^m P(k_{c_{n-1}}) \right) P(k_l) \right)$$

The optimal code-rate assignment for any given set of chunks can then be found by using the same constrained optimization strategy as in the single chunk case, with the modified utility expression above. In particular, we search for the smallest value of  $\lambda$  such that  $M^{(\lambda)} \leq M_{\text{max}}$ , where  $M^{(\lambda)}$  is the total transmission length associated with the code-rates,  $k_i^{(\lambda)}$ , which maximize

$$U^{(\lambda)} - \lambda M^{(\lambda)} = \sum_{m=1}^C \sum_{l=c_m}^{c_{m+1}-1} \left( U_{\text{mod}_l} P(k_i^{(\lambda)}) - \lambda M_l R(k_i^{(\lambda)}) \right)$$

This new functional is more difficult to optimize than that in equation (2), since the product terms in  $U_{\text{mod}_l}$  couple the impact of code-rate assignments for different layers. In fact, the optimization objective is generally multi-modal, exhibiting multiple local optima. Nevertheless, it is possible to devise a simple optimization scheme which rapidly converges to one of these local optima, with good results in practice. Specifically, given an initial set of code-rate assignments, and considering only one layer,  $\mathcal{L}_l$ , at a time, we may find the value of  $k_l$  which maximizes  $U - \lambda M$ , subject to all other code-rates being kept constant. The local optimization step for layer  $\mathcal{L}_l$  involves maximizing a quantity,  $U_{\text{eff}_l} P(k_l) - \lambda M_l R(k_l)$ , where

$$U_{\text{eff}_l} = U_l \prod_{m=2}^{m'} P(k_{c_{m-1}}) + \delta \quad (3)$$

and

$$\delta = \begin{cases} 0 & l \neq c_{m'+1} - 1 \\ \sum_{m=m'}^C \sum_{l=c_m}^{c_{m+1}-1} U_l P(k_l) & l = c_{m'+1} - 1 \\ \prod_{n=2, n \neq m'+1}^{m'} P(k_{c_{n-1}}) & l = c_{m'+1} - 1 \end{cases}$$

These local optimization problems are thus identical to those encountered in the single chunk case, replacing each layer's utility with the effective utility,  $U_{\text{eff}_l}$ .

Since each local optimization step either increases the objective,  $U - \lambda M$ , or leaves it unchanged, and this objective is necessarily bounded above, the algorithm must eventually converge as we cycle through each layer in turn, adjusting its code-rate while holding the others constant.

Multiplying probability values in the effective utility equation is essentially a multiplication of convex functions. Since a product of convex functions is non-convex, it can be shown that for any given  $\lambda$ , the objective function is generally non-convex; in

fact, there are generally multiple local optima. There is no guarantee that the result to which the code-rate assignment algorithm converges is the global optimum. Nevertheless, in practice the algorithm appears to yield good results for short code lengths, as we shall see later in Section 5.

We are still left with the problem of determining the best assignment of layers to chunks. Our algorithm for chunk assignment starts by assuming that all source layers can fit into a single chunk, which we term a "virtual chunk." After finding the optimal code-rate assignment for this single virtual chunk, all source layers are grouped into  $C$  chunks,  $\{\mathcal{C}_1, \dots, \mathcal{C}_C\}$ , whose sizes are determined by the packet size,  $S$ , and the code length,  $N$ . Effective utilities,  $U_{\text{eff}_l}$ , for all source layers are set according to equation (3) and one layer is chosen to be a pivot, designated as  $\mathcal{P}$ . A pivot is a layer, which is not assigned the strongest code,  $(N, 1)$ , but its preceding layers are assigned  $(N, 1)$  code.

Firstly, we start by holding the code-rate of the pivot layer,  $R(k_{\mathcal{P}})$ , constant and set  $\lambda = \frac{U_{\text{eff}_{\mathcal{P}}} P(k_{\mathcal{P}})}{M_{\mathcal{P}} R(k_{\mathcal{P}})}$ . Applying  $\lambda$  to other layers will cause them to converge to a new set of codes,  $\{k_l^{(\lambda)}, l \neq \mathcal{P}\}$ . These are valid solutions because the total length,  $M^{(\lambda)}$ , is less than the previous total length and so will never exceed the maximum length,  $M_{\text{max}}$ . Now, suppose that we increase the pivot's code-rate by assigning a stronger code and hence, reduce  $\lambda$ . Then, other layers will be assigned another new set of stronger codes and the new  $M^{(\lambda)}$  is higher. Repeating this process will eventually bring  $M^{(\lambda)}$  incrementally to  $M_{\text{max}}$  and so, the optimal solution is found. When the pivot is assigned  $(N, 1)$  code but the total length is still less than the maximum length, the next layer is chosen to be the new pivot,  $\mathcal{P} = \mathcal{P} + 1$ , and the process above continues by fixing the code-rate of the new pivot and applying the resulting  $\lambda$  to other layers. The process iterates through each successive layer until the total length reaches the maximum length. Since our problem here is discrete, it is unlikely for  $M^{(\lambda)}$  to be exactly equal to  $M_{\text{max}}$ . Therefore, when  $M^{(\lambda)} > M_{\text{max}}$ , a bisection method is employed to find a set of codes such that  $M^{(\lambda)}$  is just below  $M_{\text{max}}$ .

Convergence of the iterative joint chunk assignment and code-rate optimization process is assured by the following important observation. Incrementally increasing the code-rate of the pivot layer will cause the total length to increase monotonically. The transfer of pivot status to the next layer whenever the current layer is assigned the strongest code also increases the total length in the same fashion. This monotonic property of the iterative algorithm ensures its convergence.

## 5. RESULTS

In this section, we compare the total expected utility of the compressed image at the destination, whose code-rates have been determined using the single chunk and multiple chunk assignment strategies described in Sections 3 and 4. We select a code length of  $N = 50$ , a maximum transmission length of  $M_{\text{max}} = 10^6$  bytes, a range of bit error probability,  $\epsilon$ , and packet sizes,  $S$ . The scalable data source used in these experiments is a JPEG2000 compressed image, having  $L = 30$  quality layers. For simplicity, we assume that in the event any part of any layer is corrupted, the entire layer will be rendered useless, along with all subsequent layers which depend upon it<sup>1</sup>.

<sup>1</sup>In practice, this assumption is excessively conservative, since JPEG2000 decoders are able to recover well from some types of error.

**Table 1.** Comparative results given  $\epsilon = 10^{-4}$ .

S	p	M	PSNR	M*	PSNR*
2,000	0.202	201,000	11.1759	570,750	14.0564
500	0.67	701,920	24.5737	612,650	26.5
200	0.852	899,750	29.65	978,850	32.1842

**Table 2.** Comparative results given  $\epsilon = 10^{-5}$ .

S	p	M	PSNR	M*	PSNR*
20,000	0.202	200,670	11.663	200,670	11.663
10,000	0.45	662,900	23.60	666,150	23.7457
2,000	0.85	899,700	32.029	922,750	32.1
1,000	0.923	792,650	31.1991	818,300	32.154
500	0.96	741,450	32.2349	882,500	33.5109

If all of the coded source layers are able to fit inside a single chunk, subject to the constraints determined by  $N$  and  $S$ , the single chunk code-rate assignment will be optimal. Otherwise, source layers must be broken into multiple chunks, violating some of the assumptions underlying the single chunk assignment strategy. In this case, we would expect to see the multiple chunk assignment strategy providing superior performance. The results shown in Table 1 and 2 confirm this suspicion. In the tables, the resulting total length and PSNR values from the multiple chunk assignment strategy are marked with asterisks. The utility measure used here is negated MSE, and the total expected utility is conveniently expressed in terms of PSNR<sup>2</sup>. An improvement in the expected utility is equivalent to an increase in PSNR.

Table 1 and 2 show that both single and multiple chunks assignment strategies produce higher PSNR when the packet sizes are small. This owes to the fact that for a given  $\epsilon$ , the recovery probability,  $p$ , increases with decreasing packet size. High recovery probability allows the layers to be assigned with weak codes and hence, the layers are encoded with low code-rates. Therefore, it is possible to transmit more encoded layers without exceeding the maximum length,  $M_{\max}$ .

The PSNR values from the multiple chunk assignment strategy are higher relative to the single chunk case. The chunking process, described in Section 4, increases the effective utilities of layers in the upper chunks relative to those of layers in the lower chunks, which causes the same or a stronger code to be assigned to the upper chunks. As a result, the layers in the upper chunks are not corrupted as easily by packet loss.

With small packet sizes, the multiple chunk assignment strategy assigns zero code-rate to lower chunks, which means that they are not even transmitted, because the upper chunks are assigned stronger codes. Hence, they will generally survive the packet loss and produce higher PSNR. Meanwhile, the single chunk assignment strategy still assigns weak codes to the lower chunks.

## 6. CONCLUSIONS

Although PET provides an excellent framework for optimal protection of scalable data sources against loss, it suffers from the difficulty that all channel codes must span the entire collection

<sup>2</sup>PSNR (Peak Signal to Noise Ratio) is defined as  $10 \log_{10} (P^2/\text{MSE})$ , where  $P$  is the peak-to-peak signal amplitude. In this case,  $P = 255$ , since we are working with 8-bit images.

of network packets. In many practical applications, the amount of source data is large and the packet size small, leading to extremely long and computationally demanding channel codes. Two solutions to this problem present themselves immediately. Smaller network packets can be aggregated into larger packets, thereby reducing the lengths of the channel codes. Unfortunately, an erasure channel model is required so that the larger aggregate packets must be considered lost if any of their constituent packets are lost. Clearly, such a solution is inappropriate for channels with significant packet loss probability.

As an alternative, the code-rate assignment optimized for the PET framework could be used with smaller channel codes, representing smaller chunks of the total transmitted data. When the data must be divided up into independently coded chunks with smaller code lengths, the multiple chunk assignment strategy is able to provide significant improvements in expected utility (PSNR). Nevertheless, the need to use multiple chunks does impose a significant performance penalty of its own.

One drawback of the multiple chunk assignment strategy used here is amount of computation involved in determining the assignment at the transmitter. We are actively pursuing optimization schedules to reduce the computation effort. This is particularly important for a large number of source layers and/or small chunk sizes. It is reasonable to expect exactly these conditions when transmitting a large compressed image over a wireless network. We are also currently investigating modifications to the proposed encoding scheme for use with data sources having more complex dependencies.

## 7. REFERENCES

- [1] J. Shapiro, "An embedded hierarchical image coder using zerotrees of wavelet coefficients," *Proc. IEEE Data Compression Conf. (Snowbird)*, pp. 214–223, 1993.
- [2] A. Said and W. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circ. Syst. for Video Tech.*, pp. 243–250, June 1996.
- [3] ISO/IEC 15444-1, "Information technology – JPEG 2000 image coding system – Part 1: Core coding system," 2000.
- [4] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Trans. Image Proc.*, vol. 9, pp. 1158–1170, July 2000.