APPROXIMATELY OPTIMAL ASSIGNMENT FOR UNEQUAL LOSS PROTECTION

Alexander E. Mohr, Richard E. Ladner

Box 352350
Department of Computer Science and Engr.
University of Washington
Seattle, WA 98195-2350

Eve A. Riskin

Box 352500
Department of Electrical Engr.
University of Washington
Seattle, WA 98195-2500

ABSTRACT

This paper describes an algorithm that achieves an approximately optimal assignment of forward error correction to progressive data within the unequal loss protection framework [1]. It first finds the optimal assignment under convex hull and fractional bit allocation assumptions. It then relaxes those constraints to find an assignment that approximates the global optimum. The algorithm has a running time of O($hNlogN$) where h is the number of points on the convex hull of the source's utility–cost curve and N is the number of packets transmitted.

1. INTRODUCTION

The Internet is a widely deployed network of computers that exchange data packets. In traversing the network, a packet is sent from computer to computer until it arrives at its destination. However, when the number of packets sent exceeds transmission capacity, packets are discarded at random, causing loss of data and perhaps decoding failure if the lost data are not somehow replaced.

The standard practice for many networks is to request retransmission of the lost data, which requires transmitting a message back to the sender and an additional delay while the receiver awaits the retransmitted data. However, the receiver may not want to transmit a message to the sender (for example, to conserve power, to avoid broadcasting its location, or to avoid feedback implosion in multicast) or the added delay may be long (for example, satellite or transoceanic links).

Unequal loss protection [1] avoids retransmission by applying forward error correction to the source data and using that redundancy to reconstruct lost data. It uses various strengths of Reed-Solomon block codes [2] to protect the source data unequally: important data are assigned stronger codes and less important data are assigned weaker codes. In [1], we presented the unequal loss protection framework and an assignment heuristic that finds a local optimum that maximizes the benefit to the receiver.

Supported in part by National Science Foundation Grant EIA-9973531, the Washington Technology Center, and AT&T Bell Laboratories.

In this work, we introduce a new assignment algorithm that approximates the global optimum, has a time bound linear in the length of the input data, and executes multiple orders of magnitude faster than both our previous heuristic and an algorithm by Puri and Ramchandran [3].

2. BACKGROUND

This section first overviews our assumptions about the source data and introduces the concept of a utility function over that data. After discussing network loss estimation, Reed-Solomon codes are briefly reviewed. Finally, we explain how all of these elements are used within the unequal loss protection framework.

2.1. Source Data and Utility–Cost Functions

We assume that the source data are a continuous sequence of symbols. Furthermore, we assume that the data have associated with them functions that describe both their utility and cost. The utility measures how much benefit the receiver is likely to enjoy from receiving and decoding the data. The cost measures how much we must pay to achieve a certain utility and is usually given in bits.

As a simple example, if the sender knows nothing about the utility of the source data, then the utility function would be linear with cost: the sender has no information on which to base its decision. The utility values might also be arbitrary: the sender knows only that some data have a "high priority" designation and other data do not. Alternatively, if the data represent an image, the sender could choose a measure of image quality as the utility value. That measure might be the signal-to-noise ratio of the image, resulting in a utility–cost curve that tracks the PSNR–rate curve of the source. Utility might also be measured as the decrease in mean squared error or some perceptual criterion.

We assume that the utility–cost function is given by the source. Furthermore, we assume that the data are progressive: data with high utility–cost slopes generally tend to appear early in the sequence, while data with low utility–cost slopes generally tend to appear late in the sequence. This assumption allows us to approximate the utility–cost curve by its upper convex hull.

2.2. Network Loss Estimation

Another input to our algorithm will be an estimate of current packet loss conditions on the network. For N fixed, define a discrete random variable X as the number of packets out of N received, with PMF $p_X(x)$. This estimate could be the result of almost any model of expected packet loss rates: uniform, binomial, Zipf, Poisson, exponential, Gilbert–Elliott, and other distributions. The estimate could also be generated by a dynamic estimator of network conditions.

Characterizing networks such as the Internet is an open and active research topic in the networking community [4, 5]. By stipulating that our algorithm input only a PMF, we maintain the relevance of our assignment algorithm to a variety of network estimators.

2.3. Reed-Solomon Codes

Systematic Reed-Solomon (R-S) codes can be used to generate forward error correction (FEC); R-S codes are particularly effective at recovering from erased symbols when the locations of the erased symbols are known. Because we are concerned only with networks in which packets either arrive perfectly intact or are completely discarded, we consider R-S codes that are optimized for erasures and do not correct bit errors [2]. These maximum distance separable block codes are denoted by a pair (N, k), where N is the block length and k is the number of source symbols. When the code is systematic, the first k of the N encoded symbols are the source symbols, and the remaining $N - k$ symbols are redundancy. They have the property that an (N, k) code can exactly recover the k source symbols from any size k subset of the N total symbols. This recovery is possible by treating the source symbols as the coefficients of a polynomial in a Galois field and evaluating it at a number of additional points, thus creating redundant data.

2.4. Unequal Loss Protection Framework

The unequal loss protection (ULP) framework [1] was inspired by work on Priority Encoding Transmission [6]. If there are N packets of length L symbols, the basic concept is to use the ith symbol of every packet to create the ith code block, for a total of L independent R-S code blocks with various strengths. For a block containing both data symbols and FEC symbols, so long as the number of lost packets is less than or equal to the number of FEC symbols, the entire block can be decoded [6]. Note that each block will lose exactly the same number of symbols because one lost packet erases one symbol from every block.

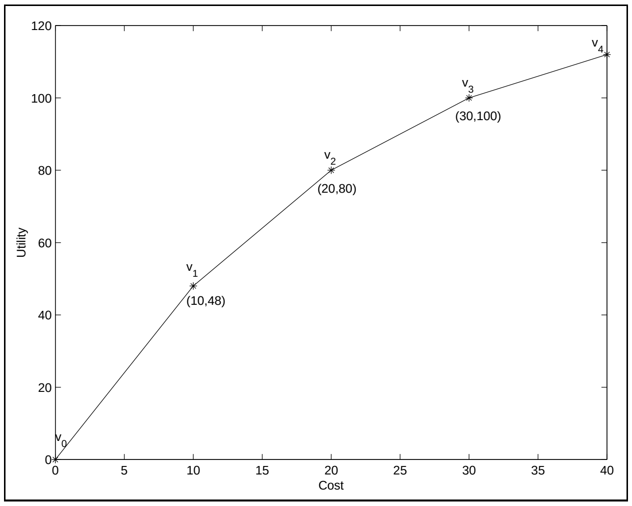

Fig. 1. An example utility–cost curve with vertices labeled.

The optimization problem can now be stated: for a given input data sequence, utility function, and PMF of packet losses, how many data symbols should be used and how strong should the R-S code on each data symbol be such that the expected received utility is maximized for a fixed cost? In the next section, we address that question.

3. APPROXIMATELY OPTIMAL ASSIGNMENT

The approximately optimal assignment algorithm consists of three distinct stages. The first stage finds the upper convex hull of the utility–cost function for the source data. The second stage assumes that R-S codes can be allocated fractionally and calculates a globally optimal assignment of forward error correction under that assumption. The final stage removes the assumptions and converts that assignment into an approximately optimal one.

3.1. Convex Hull

A number of techniques exist for finding the convex hull of a set of points [7, 8]. In our case, n input points are sorted in the cost coordinate so that a simple $O(n)$ algorithm can be used to generate the $h \leq n$ points on the upper convex hull, an example of which appears in Figure 1.

Iteration	$k=3$, a $(3,3)$ code	$k=2$, a $(3,2)$ code	$k=1$, a $(3,1)$ code	C
1	v_0	v_0	v_0	0
	$v_0 \to v_1$: $\Delta U_3 = (48-0)(0.25) = 12$ $\Delta C_3 = (10-0)(\frac{3}{3}) = 10$ $\frac{\Delta U_3}{\Delta C_3} = \frac{12}{10}$	$v_0 \to v_1$: $\Delta U_2 = (48-0)(0.50) = 24$ $\Delta C_2 = (10-0)(\frac{3}{2}) = 15$ $\boxed{\frac{\Delta U_2}{\Delta C_2} = \frac{24}{15}}$	$v_0 \to v_1$: $\Delta U_1 = (48-0)(0.75) = 36$ $\Delta C_1 = (10-0)(\frac{3}{1}) = 30$ $\frac{\Delta U_1}{\Delta C_1} = \frac{36}{30}$	
2	v_1	v_1	v_0	15
	$v_1 \to v_2$: $\Delta U_3 = (80-48)(0.25) = 8$ $\Delta C_3 = (20-10)(\frac{3}{3}) = 10$ $\frac{\Delta U_3}{\Delta C_3} = \frac{8}{10}$	$v_1 \to v_2$: $\Delta U_2 = (80-48)(0.50) = 16$ $\Delta C_2 = (20-10)(\frac{3}{2}) = 15$ $\boxed{\frac{\Delta U_2}{\Delta C_2} = \frac{16}{15}}$	$v_0 \to v_1$: $\Delta U_1 = (48-0)(0.75-0.50) = 12$ $\Delta C_1 = (10-0)(\frac{3}{1}-\frac{3}{2}) = 15$ $\frac{\Delta U_1}{\Delta C_1} = \frac{12}{15}$	
3	v_2	v_2	v_0	30
	$v_2 \to v_3$: $\Delta U_3 = (100-80)(0.25) = 5$ $\Delta C_3 = (30-20)(\frac{3}{3}) = 10$ $\frac{\Delta U_3}{\Delta C_3} = \frac{5}{10}$	$v_2 \to v_3$: $\Delta U_2 = (100-80)(0.50) = 10$ $\Delta C_2 = (30-20)(\frac{3}{2}) = 15$ $\frac{\Delta U_2}{\Delta C_2} = \frac{10}{15}$	$v_0 \to v_1$: $\Delta U_1 = (48-0)(0.75-0.50) = 12$ $\Delta C_1 = (10-0)(\frac{3}{1}-\frac{3}{2}) = 15$ $\boxed{\frac{\Delta U_1}{\Delta C_1} = \frac{12}{15}}$	
4	v_2	v_2	$\frac{2}{3}v_1$	40

Table 1. Illustration of Assignment Algorithm

3.2. Assignment Algorithm

Let $\{v_0, v_1, \ldots, v_{h-1}\} = \{(c_0, u_0), (c_1, u_1), \ldots, (c_{h-1}, v_{h-1})\}$ be the h vertices of the convex hull of the utility–cost curve for the source data, where c_i is the cost of transmitting the data that results in a utility of u_i. Assume that the utility–cost curve is convex, $\frac{u_{i+2}-u_{i+1}}{c_{i+2}-c_{i+1}} \le \frac{u_{i+1}-u_i}{c_{i+1}-c_i}$ for $0 \le i < h-2$, and that v_0 has $c_0 = 0$. In the following discussion, we refer to vertices as being protected or decoded, meaning that the source data represented by the cost/utility pair of that vertex is being protected or decoded.

If $U(x)$ represents the utility achieved when x packets are received, then the expected utility $E[U]$ of the receiver can be expressed as $E[U] = \sum_{x=0}^{N} p_X(x)\ U(x)$. For a given utility–cost curve, number of packets to transmit N, network estimate $p_X(x)$, and transmit cost C_{max}, our goal is to maximize $E[U]$.

We assume initially that a dummy vertex v_0 with zero utility and cost can be decoded no matter how many packets are received. The total cost for that assignment is $C = 0$ and the expected utility is $U = 0$. Thereafter, we examine each possible value of X and consider the effect of allowing an additional vertex to be decoded. If v_i is the highest-numbered vertex that can be decoded when k packets are received, then by assigning the next highest vertex v_{i+1} to an (N, k) code, the resulting changes to U and C are:

$$\Delta U_k = (u_{i+1} - u_i) \sum_{j=k}^{N} p_X(j). \quad (1)$$

$$\Delta C_k = (c_{i+1} - c_i)(\frac{N}{k}). \quad (2)$$

These equations assume that v_{i+1} had never before been assigned a code. If it had previously been assigned a weaker (N, m) code, then the changes that result from promoting it to a stronger (N, k) code would be:

$$\Delta U_k = (u_{i+1} - u_i) \sum_{j=k}^{m-1} p_X(j). \quad (3)$$

$$\Delta C_k = (c_{i+1} - c_i)(\frac{N}{k} - \frac{N}{m}). \quad (4)$$

Our algorithm is then as follows:

1. Calculate ΔU_k and ΔC_k for $1 \le k \le N$.
2. Choose k to maximize the ratio $R_k = \frac{\Delta U_k}{\Delta C_k}$.
3. If $C + \Delta C_k < C_{max}$, then assign v_{i+1} an (N, k) code, update C, and go to step 1.

4. Otherwise, linearly interpolate an assignment of (N, k) between v_i and v_{i+1} such that $C = C_{max}$ and terminate.

The number of iterations of the algorithm is bounded by hN because each vertex can decrease its assignment at most N times. The time of each iteration is $O(N)$ time to find the k with maximal R_k. Hence the overall complexity of the algorithm is $O(hN^2)$. That running time can be improved, however, by using a priority queue in step 2. With that modification, the complexity of the algorithm is $O(hNlogN)$.

In Table 1, we illustrate how our algorithm finds an assignment for the utility–cost curve of Figure 1; $N = 3$; a uniform PMF $p_X(x) = 0.25, x = 0, 1, 2, 3$; and $C_{max} = 40$. The resulting assignment is a $(3, 1)$ code protecting the first $\frac{2}{3}$ of source data in v_1 and a $(3, 2)$ code protecting the last $\frac{1}{3}$ of the source data in v_1 and all of the source data between v_1 and v_2. Source data after v_2 would not be sent.

3.3. Resolving the Assignment

To find an actual assignment, we need to remove the fractional bit assumption. Because the assignment found above is optimal only for that assumption, this step makes the actual assignment only approximately optimal. Nevertheless, we use the optimal solution as a guide and note that because of the convexity requirement, each of the R-S codes has a number of decodable source symbols that is non-decreasing as the code strengths decrease. Thus, we start with the strongest code, $(N, 1)$, and allocate enough code blocks so that the number of source symbols protected equals or exceeds the number suggested by the guide. We then move to the next-strongest code and repeat. This stage uses O(n) steps.

4. RESULTS

We implemented the $O(hN^2)$ version of our algorithm and compared it to our previous algorithm [1]. When finding an assignment for the 512×512 Lenna image using 137 packets of size 47 bytes, a total rate of 0.2 bits per pixel, and an exponential loss model with 20% mean loss rate, we found an assignment in 75 ms on a SPARCstation 10, two orders of magnitude faster than the 7.3 s of the previous algorithm. Execution time dropped to 7.6 ms on a 500 MHz Intel Pentium III workstation. The assignment resulted in the same expected PSNR of 29.42 dB, which is within 0.06 dB of an upper bound on the global optimum.

We also encoded slightly more than 10 seconds of the 176×144 Foreman video sequence at 1.0 bits per pixel with 3D-SPIHT, generating 811 packets of 1000 bytes each. With a 5% mean loss rate, our SPARCstation 10 was able to find an assignment in 0.47 seconds.

In related work, Puri and Ramchandran have also developed an approximately optimal assignment algorithm for unequal loss protection that uses a Lagrangian multiplier λ [3]. Performance results for our two assignment algorithms are nearly identical (to within measurement error), but our algorithm executes over 50 times faster than their reported results. However, in communications with them, we have learned that their execution time includes file parsing, whereas ours does not, so the actual performance difference is unclear. Furthermore, although their algorithm is linear in the number of input points, they do not bound the number of iterations required to find a value for λ.

5. CONCLUSION

In this paper, we have presented a fast algorithm that approximates the optimal assignment of unequal amounts of Reed-Solomon codes to progressive data. The algorithm has explicit bounds on running time and executes extremely quickly in practice, and thus should make real-time assignment of FEC data possible.

6. REFERENCES

[1] Alexander E. Mohr, Eve A. Riskin, and Richard E. Ladner, "Graceful degradation over packet erasure channels through forward error correction," in *Proceedings of Data Compression Conference*, Mar. 1999, pp. 92–101.

[2] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM Computer Communication Review*, vol. 27, no. 2, pp. 24–36, Apr. 1997.

[3] Rohit Puri and Kannan Ramchandran, "Multiple description source coding through forward error correction codes," in *Proc. 33rd Asilomar Conference on Signals, Systems, and Computers.*, Oct. 1999.

[4] Stefan Savage, "Sting: a TCP-based network measurement tool," in *Proceedings of the 1999 USENIX Symposium on Internet Technologies and Systems*, Oct. 1999, pp. 71–79.

[5] Vern Paxson and Sally Floyd, "Why we don't know how to simulate the Internet," under review, Oct. 1999.

[6] Andres Albanese, Johannes Blömer, Jeff Edmonds, Michael Luby, and Madhu Sudan, "Priority encoding transmission," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1737–1744, Nov. 1996.

[7] R. A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane," *Information Processing Letters*, vol. 2, pp. 18–21, 1973.

[8] D. G. Kirkpatrick and R. Seidel, "The ultimate planar convex hull algorithm?," *SIAM Journal on Computing*, vol. 15, no. 2, pp. 287–299, 1986.