Coding for Parallel Links to Maximize Expected Decodable-Message Value

Christopher S. Chang
Department of Electrical Engineering
California Institute of Technology
Pasadena, CA 91125, USA
E-mail: cswchang@caltech.edu

Matthew A. Klimesh
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109, USA
E-mail: Matthew.A.Klimesh@jpl.nasa.gov

***Abstract*—Future communication scenarios for NASA spacecraft may involve multiple communication links and relay nodes, so that there is essentially a network in which there may be multiple paths from a sender to a destination. The availability of individual links may be uncertain. In this paper, scenarios are considered in which the goal is to maximize a payoff that assigns weight based on the worth of data and the probability of successful transmission. Ideally, the choice of what information to send over the various links will provide protection of high value data when many links are unavailable, yet result in communication of significant additional data when most links are available. Here the focus is on the simple network of multiple parallel links, where the links have known capacities and outage probabilities. Given a set of simple inter-link codes, linear programming can be used to find the optimal timesharing strategy among these codes. Some observations are made about the problem of determining all potentially useful codes, and techniques to assist in such determination are presented.**

I. Introduction

When communicating through a noisy one-way communications link, it is well known that it is often practical to achieve a high level of reliability by protecting the communicated data with error-correcting codes, provided the attempted data rate is not too high. However, suppose that with some appreciable probability the communications link may fail for the entire duration of the communication attempt. Clearly, error-correcting codes are of no use in protecting against this type of link failure. But suppose there are more than one of these unreliable links. Then the communicated data can be protected somewhat with error-correcting codes applied between the multiple links. A trivial example of what we mean by this is if there are three links and identical data is sent on all links, thus protecting against failure of any two links (but not all three links).

More generally, we may want to send multiple messages with different worths (or priorities) from one point to another in a general network that contains unreliable links. Ideally, we would want to achieve the maximum available throughput at all times, in a way that: (1) protects higher value data, and (2) does not require prior knowledge of the network state. Usually this ideal goal will not be achievable, but we can consider tradeoffs. Roughly speaking, we would like to provide protection of higher value data when a large fraction of the links in a network are unavailable, and to achieve transmission of significant additional data when most links are working properly.

In this paper we restrict our attention to one simple type of network: a single source node and the single destination node connected by parallel unreliable links. We postulate that it is reasonable to model (or at least approximate) some communications scenarios as communications through this type of network. We develop and analyze a model whose main feature is communication through parallel unreliable links that do not change state (between working and non-working) for the duration of the communication attempt. Some additional key features of our model are:

- a given link fails with some known probability, otherwise the link provides reliable communications with a known capacity;
- link failures are independent;
- the sender does not know the status of the links;
- there are some number of messages to be sent, and the messages have known worths (i.e., priorities or values) and sizes;
- the worth of a partial message is proportional to its size (i.e., partial credit is given for partial messages); and
- the payoff of a given link usage strategy is the expected total value of the messages successfully decoded.

The following (not entirely realistic) example scenario falls under our model (see Fig. 1). Consider a rover on Mars that, in a given period of time, can communicate to Earth in three ways: a direct-to-Earth link and two different relays through spacecraft orbiting Mars. It is assumed that these links have independent and non-negligible outage probabilities. It is also assumed that the resources (e.g., time and power) needed to utilize these links are small, so that the rover should always attempt to send information through all three routes. Then, given a list of messages and the values of their being received during this time period, we would like to find the best combinations of messages to send on each of the three

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

978-1-4244-7417-2/10/$26.00 ©2010 IEEE

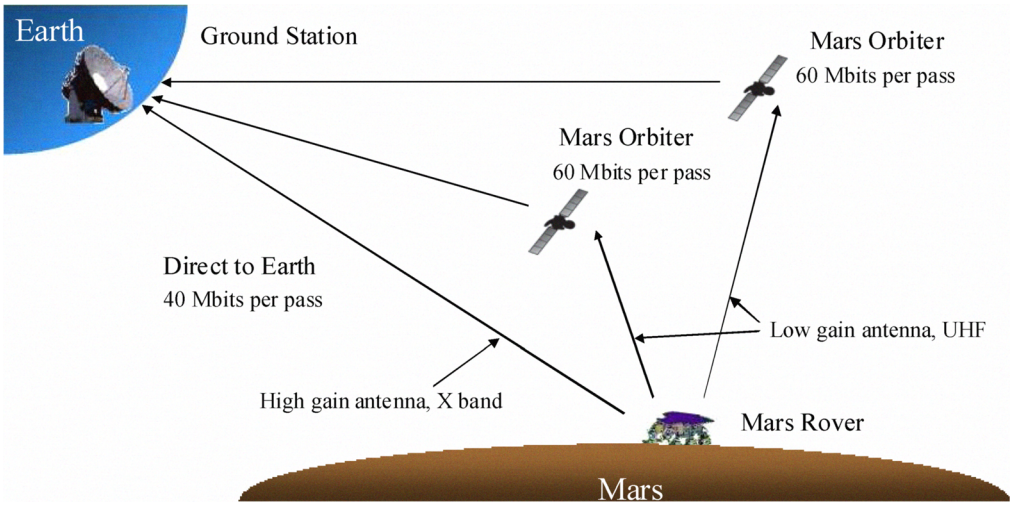

Fig. 1. Possible application: a rover on Mars attempts to communicate to Earth by three independent paths.

links. Note that the separate links do not have to be used simultaneously to fit our model. In fact, it is possible to use our model to represent a single link used at different times, provided the outage probabilities can still be modeled as being independent.

We will present the model as though a given link capacity indicates the maximum amount of data that can be transmitted through the link (if it is up) during the communication attempt, and message sizes are also amounts of data on the same scale. Thus in a sense the communication attempt is an one-shot action. However, it would be equally valid to regard both capacities and message sizes as representing data rates in bits per unit time. From this viewpoint the model would pertain to communication for an indefinite period of time, during which it is not possible to inform the sender which links are working.

Communication strategies for our model involve time-sharing among relatively simple inter-link codes. Given a collection of candidate inter-link codes, the optimal timesharing proportions among them can be determined using linear programming. However, it appears to be difficult to determine the set of all "useful" inter-link codes for a given number of messages and links. We discuss this problem and describe an algorithm that can assist in determining all potentially useful inter-link codes that use a subset of the links and use these links equally, for a given number of messages and links.

In most communications scenarios previously studied, one is either interested in achieving arbitrarily small loss or error rates (in theoretical studies), or merely very small loss or error rates (in practical systems). The communication model described here is conceptually different in that it need not deal with small loss rates. Under our model we must accept that losses will occur and we measure performance in terms of the messages successfully communicated. This model may be appropriate for, say, a spacecraft that is capable of gathering much more information than it can transmit to Earth.

When generalized to general networks, our model can be thought of as a network coding problem. Most previously considered network coding problems fall into one of two categories: either they seek to maximize (a constant) throughput in a fixed network, or they seek robust communication at a constant rate in an unreliable network. Our model differs from both of these in that the throughput will depend on the network state even though a fixed coding scheme is used.

Indeed, even though we only need codes for correcting erasures and are primarily concerned with short block lengths, our particular model appears to make investigation of useful codes challenging. Not only are we interested in recovery of partial information when the whole codeword cannot be recovered, but the different codeword positions correspond to different links, and thus can have different erasure probabilities.

A. Related Work

One somewhat related investigation of note is the Priority Encoding Transmission (PET) scheme of Albanese *et al.* [1]. Also of note is the extension of Silva and Kschischang [2] which shows how PET can be used in a network coding system. Like our model, the PET model includes a number of messages to be transmitted, and each message has an associated priority. Transmitted packets may be lost, but a given message can be recovered if a sufficient fraction of packets arrive successfully, where the fraction depends on the message's priority. One key difference between our model and the PET model is that under the PET model, there can be a large number of packets that may be received or lost independently, while under our model all packets sent on a given link are either lost or successfully received together, and there are a small number of different links. Our model includes outage probabilities, while under PET the only concern is the fraction of packets successfully received. Under PET

all information is protected with (possibly trivial) maximum distance separable (MDS) codes. Under our model MDS codes may be used, but the natural way to use them would be to put different symbols on different links. When used this way the blocks lengths would never need to be larger than the number of links. Furthermore, we are concerned with partial decoding of codes, and this turns out to imply that non-MDS codes can be useful.

If the link capacities are all equal, then under our model one could regard the union of the parallel links as a single channel with an unknown state. One symbol for this channel would be a vector consisting of one symbol for each link. (This idea could also be extended to accommodate differing link capacities.) This channel now fits the assumptions of a compound channel [3], but since all links might be down, the usual compound channel capacity (the maximum guaranteed throughput) would be zero. Aside from that, the compound channel model does not consider probabilities for the channel states or differing data worths.

Again regarding our model as a single channel with an unknown state, we observe that it is related to the broadcast channel model [4], where each channel state corresponds to a separate "receiver". The broadcast channel problem is to find the jointly achievable rate region. The broadcast channel model is closely related to the unequal error protection model (see, e.g., [5]), which includes the concept of differing data worths. As a broadcast channel, with N parallel links our model would have $2^N - 1$ receivers (disregarding the state where all links are down). Thus even for small N it appears to be unwieldy to apply general broadcast channel results to our model.

B. Preliminaries

Let N be the number of parallel links. For $i \in \{1, \ldots, N\}$, link i has capacity c_i and outage probability p_i (equivalently, success probability $\bar{p}_i = 1 - p_i$). Let M be the number of messages. For $j \in \{1, \ldots, M\}$, message j has size s_j and worth per unit size π_j. The units of the link capacities and message sizes are arbitrary (but are the same for all links and messages).

Informally, for each link i, the sender sends a stream of data that is a function of the M messages and that is not longer than the link capacity c_i. The function can depend on all of the model parameters above. The receiver successfully receives the data on some subset of the links (and this subset is known to the receiver). The receiver then reconstructs the original messages to the extent possible from the data received. Let R_j be a random variable indicating how many size units of message j are recovered. The payoff from a communication attempt is the sum $\sum_{j=1}^{M} R_j \pi_j$. The payoff from a communication strategy is the expected value of this quantity.

It is implicit in this model that the message sizes are large and thus the messages can be split into pieces closely approximating any given fraction. It is possible to more formally describe the space of communication strategies using codes on discrete alphabets and allowing limiting cases as the size unit becomes large; however, here we stick with our informal description.

C. Structure of the Paper

In Section II, we propose a linear programming formulation to maximize the payoff when we are given a set of candidate codes to timeshare among. In Section III, we consider the problem of finding a good candidate code set; specifically we present a search method for candidate codes with unit size messages and unit capacity links. Finally, Section IV concludes the paper.

II. Optimization with Linear Programming

The assumptions of our model allow us to partition the messages into pieces and combine the pieces with inter-link codes. Essentially this amounts to timesharing among different codes. Given a list of candidate codes, we can determine how to optimally timeshare among them using linear programming.

Thus "simple" candidate codes form the building blocks of a communications strategy. For convenience in presenting examples, we introduce an informal concise notation for describing candidate codes. We describe this notation with examples. Consider the case of three links and two messages ($N = 3$ and $M = 2$). We label the messages A and B. One possible code consists of sending a portion of message A on all three links; we represent this code by (A, A, A). Similarly, a code could consist of sending a portion of message B on links 1 and 3; we represent this code by $(B, -, B)$. Another possible code consists of sending the same portion of message A on link 1, an equal-sized portion of message B on link 2, and the bitwise exclusive-or of these same portions on link 3; we represent this code by $(A, B, A+B)$, where the notation $A + B$ symbolizes addition in a finite field. A code can also use two different portions of the same message, as in $(A_1, A_2, A_1 + A_2)$. Codes such as (A, A, B) that can be obtained by timesharing simpler codes (here by equal parts of $(A, A, -)$ and $(-, -, B)$) do not need to be considered as candidate codes. Although this code notation is useful for some of our purposes, we remark that in general it is not adequate to describe all possible codes.

We assume that the message sizes and message worths are all given. We also assume that the link capacities and outage probabilities are given. Suppose our list of codes contains n_c codes. Our objective is to find a column vector $\mathbf{z} = [z_1, \ldots, z_{n_c}]^T$ that describes how much we use each code.

The codes are described with an $N \times n_c$ matrix $\mathbf{K}$ that tells how much the codes use the links, an $M \times n_c$ matrix $\mathbf{L}$ that tells the message content of the codes, and an n_c element column vector $\mathbf{v}$ that gives the expected payoffs from the codes.

More specifically, in $\mathbf{K} = [\kappa_{i,k}]$ the entry $\kappa_{i,k}$ is the usage of link i by one unit of code k. In $\mathbf{L} = [\ell_{j,k}]$ the entry $\ell_{j,k}$ is the amount of message j sent with one unit of code k.

And in $\mathbf{v} = [v_1, \ldots, v_{n_c}]^T$ the entry v_k is the payoff (in expected received value) from one unit of code k; note that v_k is a function of the message worths and the link outage probabilities as well as of the properties of code k.

Observe that each code is described by an entry in $\mathbf{v}$ and a column in each of $\mathbf{K}$ and $\mathbf{L}$. As an example, consider the codes $(A, B, A+B)$ and (A_1, A_2, A_1+A_2) for the case of two messages and three links. We can describe $(A, B, A+B)$ by the columns $(1,1,1)$ and $(1,1)$ in $\mathbf{K}$ and $\mathbf{L}$ respectively. Note that the unit size is arbitrary here; we still describe the same code if we scale both columns by an arbitrary factor (and the entry in $\mathbf{v}$ would need to be scaled by the same factor). The code (A_1, A_2, A_1+A_2) can be described by $(1,1,1)$ and $(2,0)$.

With link capacity list $\mathbf{c} = (c_1, \ldots, c_N)^T$ and message size list $\mathbf{s} = (s_1, \ldots, s_M)^T$, the optimal code usage vector $\mathbf{z}$ can be determined via the following linear program:

$$\begin{aligned} \text{find } \mathbf{z} \text{ to maximize} \quad & \mathbf{v}^T\mathbf{z} \\ \text{subject to} \quad & \mathbf{K}\mathbf{z} \le \mathbf{c} \\ & \mathbf{L}\mathbf{z} \le \mathbf{s} \\ & \mathbf{z} \ge 0. \end{aligned} \tag{1}$$

As a concrete example, consider the case of 2 messages and 3 parallel links, with the following list of 17 codes:

$$\begin{aligned} \{&(A,-,-),\ (-,A,-),\ (-,-,A),\ (A,A,-),\ (A,-,A),\\ &(A,A,-),\ (A,A,A),\ (A_1,A_2,A_1+A_2),\ (B,-,-),\\ &(-,B,-),\ (-,-,B),\ (B,B,-),\ (B,-,B),\ (B,B,-),\\ &(B,B,B),\ (B_1,B_2,B_1+B_2),\ (A,B,A+B)\} \end{aligned}$$

Note that we have assumed the links are indexed so that $p_1 \le \cdots \le p_N$ (decreasing reliability) and that the messages are indexed in order of decreasing value (per unit size), $\pi_1 \ge \cdots \ge \pi_M$. With these assumptions it is easily verified that we do not need to consider any other permutations of symbols in the codes (A_1, A_2, A_1+A_2), (B_1, B_2, B_1+B_2), and $(A, B, A+B)$; for example the code $(A, B, A+B)$ is always at least as good as $(B, A+B, A)$.

For this list of codes we have the following constants $\mathbf{K}$, $\mathbf{L}$, and $\mathbf{v}$:

$$\mathbf{K} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 & \frac{1}{2} & 1 & 0 & 0 & 1 & 1 & 0 & 1 & \frac{1}{2} & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & \frac{1}{2} & 0 & 1 & 0 & 1 & 0 & 1 & 1 & \frac{1}{2} & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & \frac{1}{2} & 0 & 0 & 1 & 0 & 1 & 1 & 1 & \frac{1}{2} & 1 \end{pmatrix}$$

$$\mathbf{L} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{v}: \begin{cases} [v_1, v_9] &= [\pi_A, \pi_B]\ \bar{p}_1 \\ [v_2, v_{10}] &= [\pi_A, \pi_B]\ \bar{p}_2 \\ [v_3, v_{11}] &= [\pi_A, \pi_B]\ \bar{p}_3 \\ [v_4, v_{12}] &= [\pi_A, \pi_B]\ (\bar{p}_1 + \bar{p}_2 - \bar{p}_1\bar{p}_2) \\ [v_5, v_{13}] &= [\pi_A, \pi_B]\ (\bar{p}_1 + \bar{p}_3 - \bar{p}_1\bar{p}_3) \\ [v_6, v_{14}] &= [\pi_A, \pi_B]\ (\bar{p}_2 + \bar{p}_3 - \bar{p}_2\bar{p}_3) \\ [v_7, v_{15}] &= [\pi_A, \pi_B]\ (\bar{p}_1 + \bar{p}_2 + \bar{p}_3 - \bar{p}_1\bar{p}_2 - \bar{p}_1\bar{p}_3 - \bar{p}_2\bar{p}_3 \\ &\qquad + \bar{p}_1\bar{p}_2\bar{p}_3) \\ [v_8, v_{16}] &= [\pi_A, \pi_B]\ (0.5\bar{p}_1(1-\bar{p}_2)(1-\bar{p}_3) \\ &\qquad + 0.5\bar{p}_2(1-\bar{p}_1)(1-\bar{p}_3) \\ &\qquad + \bar{p}_1\bar{p}_2 + \bar{p}_1\bar{p}_3 + \bar{p}_2\bar{p}_3 - 2\bar{p}_1\bar{p}_2\bar{p}_3) \\ v_{17} &= \pi_A\bar{p}_1(1-\bar{p}_2)(1-\bar{p}_3) + \pi_B\bar{p}_2(1-\bar{p}_1)(1-\bar{p}_3) \\ &\qquad + (\pi_A + \pi_B)\,(\bar{p}_1\bar{p}_2 + \bar{p}_1\bar{p}_3 + \bar{p}_2\bar{p}_3 - 2\bar{p}_1\bar{p}_2\bar{p}_3) \end{cases}$$

For example, $v_4 = \pi_A(\bar{p}_1 + \bar{p}_2 - \bar{p}_1\bar{p}_2)$ because when one unit of the fourth code is used, one unit of message A can be decoded (with worth π_A) whenever at least one of the first two links are up (which happens with probability $\bar{p}_1 + \bar{p}_2 - \bar{p}_1\bar{p}_2$).

We solved the linear programming problem for a number of randomly generated scenarios (random message sizes and worths, and random link capacities and probabilities). We found that for each of the 17 codes, there were scenarios in which the code was needed in the optimal solution. The list of 17 codes can be verified to include all needed codes that use a subset of the links equally. For this relatively simple case, we think it is likely that no codes that use links non-uniformly are needed, i.e., we think the solutions obtained with these 17 codes would still be optimal if the candidate code list could be enlarged to include all possible codes.

The linear programming method clearly has limitations for our problem. For one thing, making the list of candidate codes looks to be a very complicated problem in general. In addition, the number of candidate codes grows at least exponentially with the number of links, since there are already $2^N - 1$ possibilities just for repetition codes for the first message. However, the linear programming method appears to be viable for small numbers of links, which is a case that may be of interest for some applications.

III. Unit Size and Unit Capacity Problem

In this section, we consider a special case of the problem with unit message sizes and unit link capacities. The motivation for this is try to characterize, or at least be able to generate, complete sets of candidate codes for the general case. Here we consider only codes that use all links equally (not just a subset). We include codes that are formed by timesharing codes that use disjoint sets of links. We continue to allow arbitrary link outage probabilities and message worths.

Given a list of codes, we can generate a number of random outage probabilities and message worths, and in each case check which code in the list is best. In this way we can accumulate a smaller list of codes that can be best in some scenario, and we can attempt to characterize the properties of these codes. Generating the original list of codes is formidable challenge, since even for small number of links there are an enormous number of possible codes. We could restrict our attention to linear codes, but the number of such codes is still huge. To make the problem somewhat tractable, we can instead consider possible code properties instead of explicit codes. By code properties we mean a description of which messages can be decoded when any given erasure pattern occurs. To do this, we take advantage of a connection between entropy and matroids.

For our purposes, we need only the following characterization of matroids, which is Corollary 1.3.4 in [6]:

Corollary 1: Let E be a set. A function $r : 2^E \to \mathbb{Z}^+ \cup \{0\}$ is the rank function of a matroid on E if and only if r satisfies the following conditions:

(R1) If $X \subseteq E$, then $0 \leq r(X) \leq |X|$.
(R2) If $X \subseteq Y \subseteq E$, then $r(X) \leq r(Y)$.
(R3) If X and Y are subsets of E, then $r(X \cup Y) + r(X \cap Y) \leq r(X) + r(Y)$.

Now, we consider the connection between matroids and codes in the following subsection.

A. Codes and Matroids

Suppose we have a code C for M messages and N links, say $C = (X_1, \ldots, X_N)$ where each X_i is a function of $\mathbf{W} = (W_1, \ldots, W_M)$ and the W_j's are messages of equal size. We assume sizes are normalized so that the message sizes are all 1.

Suppose now that the messages, $W_1, \ldots, W_M$, are each random variables, independent and uniformly distributed on their possible values. Then $X_1, \ldots, X_N$ are also random variables since they are functions of $\mathbf{W}$. Let $U_1 = \{W_1, \ldots, W_M\}$, $U_2 = \{X_1, \ldots, X_N\}$, and $U = U_1 \cup U_2$. We consider the joint entropies of subsets of U. For convenience, we normalize the entropies so that any message W_j has unit entropy, i.e., $H(W_j) = 1$.

Now define $f : 2^U \to [0, \infty)$ by $f(S) = H(S)$, where 2^U is the power set of U and $H(S)$ is the joint entropy of the members of S (with $H(\varnothing) = 0$). As is well known, the nonnegativity of conditional mutual information implies that this function is submodular, meaning $f(S_1 \cup S_2) + f(S_1 \cap S_2) \leq f(S_1) + f(S_2)$. Clearly f is also monotone, meaning $f(S_1) \leq f(S_2)$ whenever $S_1 \subseteq S_2$. These properties, along with $f(\varnothing) = 0$, mean f is by definition a *polymatroid function* [7]–[9]. The fact that entropy is a polymatroid function is well known; see e.g., [8], [9].

For simplicity, it seems reasonable to consider only codes for which f defined in this way is integer-valued. In any case, if a code produces an f that takes on noninteger but rational values, we can choose a new normalization of the message entropies to make the corresponding f take on only integer values, then the messages can be subdivided so that they have unit entropy again. The resulting code would be a code on a larger number of messages but would be essentially equivalent to the original code.

To further simplify, we will restrict ourselves to codes for which all X_i also have unit entropy (implying that the encoded symbols are "messages" of unit length). However, we think it is likely that there are codes that are "useful" and do not satisfy this condition.

With this further simplification the function f must be the rank function of a matroid (see [6], [8]), as it is integer-valued and satisfies $f(S) \leq |S|$.

Thus any code that satisfies our conditions has a corresponding matroid. However, the converse probably does not hold: it is likely that there are matroids that cannot be produced from a code as above. This is because it is known that there are matroids that are non-entropic, meaning there does not exist an ensemble of random variables with joint entropies corresponding to the matroids's rank function. In fact it is known that there are matroids that are not asymptotically entropic, which, loosely speaking, means they cannot be approximated closely by entropic polymatroids. See [8] for a more precise definition of asymptotically entropic matroids and an example of a matroid that is not asymptotically entropic (the Vámos matroid).

Our interest in matroid rank functions for codes stems from two observations. First, the matroid rank function contains complete information about the performance of the code. Second, it appears to be much easier to systematically generate all matroid rank functions for a given number of messages and links than it is to generate all codes (or code properties).

We consider obtaining information about the performance of the code from the matroid rank function. First note that we require $f(S) = |S|$ when $S \subseteq U_1$, since the messages are independent. We also require $f(\{X_i\}) = 1$. The properties of matroid rank functions imply that if $S \subseteq U_2$ and $W_j \in U_1$, then either $f(S \cup \{W_j\}) = f(S)$ or $f(S \cup \{W_j\}) = f(S) + 1$. The latter implies W_j is independent of S, and so cannot be determined from S. The former implies W_j is completely determined from S and so W_j can be recovered when the links corresponding to S are up.

We remark that our hypothesis that the messages are random and independent is only a tool for analysis; we do not require the messages to be random in an actual communications system. Clearly, if we want to be able to decode some W_j from a subset S of the code symbols, it must be necessary for this to be possible when the messages are random and independent. And if it is possible in that case, then clearly W_j must be deterministically obtainable from S no matter what the messages are.

As we mentioned earlier, some matroid rank functions may not correspond to any realizable code. However, if a code produces performance better or equal to that corresponding to any matroid rank function, we can still conclude the code must be optimal.

B. Automated Process: Generating Matroids and Calculating Performance Metrics

When we have M messages and N links, we consider a matroid with $(M+N)$ elements. By enumerating candidates for rank functions, we can count all possible matroids. Recall that the maximum rank is M and any singleton element has rank 1.

To efficiently generate all rank functions with given parameters, we use a backtracking algorithm [10]. Whenever we assign a possible value $(1, 2, \ldots, M)$ to an unassigned variable (rank function), we check the validity of that assignment by checking the conditions in Corollary 1.

Once we have a full list of all possible matroids, we can calculate the payoff for each matroid (each case) with an automated process. Recall that if $f(S) = f(S \cup \{W_j\})$, then W_j is decodable from the set of code positions (or links) corresponding to $S \subseteq U_2$. When we have N links, we need to check $2^N - 1$ combinations of links (excluding the empty set). For each combination, we check if the message

W_j is decodable with that combination or not. For each combination, we have a corresponding probability that is weighted by the worth of the message and then added to the payoff.

C. Systematic Codes

If the performance of a given code is never better than the performance of another code, then the former code can be eliminated from consideration. As an example, consider the codes $(A, B, A+B, A+C)$ and $(A, B, A+B, C)$. In $(A, B, A+B, A+C)$, message C appears only once (link 4), and that is in combination with message A. Thus it is necessary but not sufficient for link 4 to be up to recover message C, and it can be verified that $A+C$ is never useful for recovering any other message. Therefore we cannot do any worse by replacing $A+C$ with C. More generally, by this reasoning we can eliminate from consideration any code for which a message is only involved in one code position, and the message is combined with one or more other messages there.

One might conjecture that this idea could be generalized further and we need only consider *systematic* codes, which for our purposes are codes in which any message involved in the code is sent directly (not combined with other messages) on some link. (For example, (A, A, A) and $(A, B, A+B)$ are systematic codes under this definition.) However, perhaps surprisingly, this conjecture appears to be false: for 5 links and 3 messages, our preliminary results indicate that there are combinations of outage probabilities and link failure probabilities for which some permutations of the codes $(A, B, A+B, A+C, B+\alpha C)$ or $(A, C, A+B, A+C, B+\alpha C)$ produce a higher payoff than any other code that does not divide up messages. (Here the codes are in a non-binary field and α is not 1. The two codes are different because we are assuming the worths are in decreasing order starting from A.) These potential counterexamples were arrived at by enumerating what we think is all codes that could conceivably be optimal, then generating random scenarios (worths and outage probabilities) and checking which code gives the best payoff. This will be considered in more depth in Section III-D.

When generating codes (or, more accurately, matroids representing codes) using the automated process described in Section III-B, we can automatically determine if a given code is systematic with the following criterion:

(C1) For all message indices j such that $f(U_2) = f(U_2 \cup \{W_j\})$ (meaning that the message W_j is included in the code), there exists at least one link i such that $f(\{X_i, W_j\}) = 1$.

Note that this condition presumes that we have already required that $f(\{X_i\}) = 1$. If a code satisfies condition (C1), then the code is systematic.

D. Random Trial Results

We have run the described matroid search for 3 messages with 4 and 5 links. For the 4 links case, the search was completed. For this case we then randomly generated link outage probabilities and message worths, and in each case we determined which matroid achieved the best payoff. In a large number of trials, we did not encounter any case where the best matroid did not correspond to a non-systematic code. Furthermore, for every matroid that produced a maximum payoff, we could identify a corresponding code in a prior list of codes that we had suspected contained all codes that could be best.

For the 5 links case, we do not have a full list of matroids yet. However we performed the same random trial search with a manually constructed list of codes in which we attempted to include every code that seemed like it had a chance of being best in some case. It was in these trials that we found that the previously mentioned non-systematic codes could give the best payoff.

IV. Conclusions

We have presented a model for a communication scenario involving transmitting messages with different worths through an unreliable network. Our results concern a simple network that allows unicast communication over multiple parallel links.

We suggested a linear programming formulation that allows us to determine what combinations of messages to use across the multiple channels among a precomputed library of simple combinations of messages. We also described an algorithm for assisting in finding viable combinations of messages to populate this precomputed library.

To the best of our knowledge, the problem formulation we opened up in this paper is new. Even for the simple network considered, further development of our approach to the problem could be productive. Future research could also consider more complicated networks such as multicast networks, for which one could take advantage of network coding.

References

[1] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan, "Priority encoding transmission," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1737–1744, 1996.

[2] D. Silva and F. Kschischang, "Rank-Metric Codes for Priority Encoding Transmission with Network Coding," *10th Canadian Workshop on Information Theory*, pp. 81–84, 2007.

[3] I. Csiszár and J. Körner, *Information theory: coding theorems for discrete memoryless systems*. Academic Press, Inc. Orlando, FL, USA, 1982.

[4] T. Cover, "Broadcast channels," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 2–14, 1972.

[5] A. Calderbank and N. Seshadri, "Multilevel codes for unequal error protection," *IEEE Transactions on Information Theory*, vol. 39, no. 4, pp. 1234–1248, 1993.

[6] J. Oxley, *Matroid Theory*. New York, NY: Oxford University Press, 2006.

[7] S. Fujishige, *Submodular Functions and Optimization*. Amsterdam, The Netherlands: Elsevier, 2005.

[8] F. Matúš, "Two constructions on limits of entropy functions," *IEEE Transactions on Information Theory*, vol. 53, no. 1, pp. 320–330, 2007.

[9] R. Dougherty, C. Freiling, and K. Zeger, "Networks, matroids, and non-Shannon information inequalities," *IEEE Transactions on Information Theory*, vol. 53, no. 6, pp. 1949–1969, 2007.

[10] G. Brassard and P. Bratley, *Fundamentals of algorithmics*. Upper Saddle River, NJ: Prentice-Hall, Inc., 1996.