6.962: Graduate Seminar in Communications
Week 1 Report

Presenter: Won S. Yoon

Date: September 13, 2000

Paper: "Network Information Flow" by Ahlswede, Cai, Li, Yeung (Transactions on Information Theory, July 2000).

1 Overview

The authors of this paper address the problem of a single source multicasting information to multiple destinations over a network of nodes interconnected by noiseless, discrete, independent channels. Each of the intermediary nodes is allowed to decode and re-encode the information it receives before sending it along to the next node(s). It is required that every ultimate destination be able to reconstruct the same information at the same rate, with no distortion and with either arbitrarily-small or zero probability of error. The main contribution of this paper is the definition of a particular class of network coding schemes (called alpha-codes) and a coding theorem which gives necessary and sufficient conditions for the physical network to accomodate a given information rate under the decoding requirements. The authors further conjecture that this coding theorem can be extended to more general network coding methods.

The authors also discuss an extension to the problem with multiple sources and give an example of how this can be much more difficult than the single-source problem. Finally, the authors prove that even if probabilistic coding is allowed, it is not possible to achieve higher rates than by time-sharing general deterministic codes.

2 Problem Formulation

The communication network is defined by the following components:

- An information source: defined as a random variable X taking values uniformly from the set of messages $\Omega = \{1, 2, \ldots, 2^{nh}\}$. The entropy rate of the source is h bits per second.

- Edges (links, channels): assumed to be noiseless, discrete, and mutually independent. Each directed edge (i, j) (emanating from node i and terminating at node j) has capacity R_{ij} bits/sec. (Note: since channel alphabets are not explicitly specified in the paper, we assume that all channels are binary).

- Nodes: the authors define 3 types of nodes (shown in Figure 2).

  1. *Source node (supply node):* s observes the output of the source X, and has a separate encoder for each output link. The encoder for output link (s, i) maps the outcome of X to a symbol $f_{si}(X) \in \{1, 2, \ldots, \eta_{si}\}$, and then sends a binary description of $f_{si}(X)$ over the link (s, i) in n seconds.
  2. *Intermediate nodes:* each intermediate node has a separate encoder for every output link. At node k, the encoder for output link (k, i) takes as input all the information received from all input links into node k, denoted $z = \{f_{jk}\}_{j:(j,k)\in E}$, and maps this to a symbol $f_{ki}(z) \in \{1, 2, \ldots, \eta_{ki}\}$. The encoder then sends a binary description of $f_{ki}(z)$ over link (k, i) in n seconds.

3. *Sink nodes (destinations):* $t_1, \dots, t_L$ each tries to reconstruct the original message X with no distortion. Sink node t has a decoder which takes as input the information received from all input links to node t, denoted $w_t = \{f_{jt}\}_{j:(j,t)\in E}$. The decoder maps this information to an estimate of the original message, $\hat{X}(w_t) \in \Omega$.

The set of encoding and decoding functions at every node as described above is called an $(n, \{\eta_{ij}\}, h)$ *network code*. For a given network and source rate, the goal is to design a network code such that every sink can reconstruct X with arbitrarily-small or zero probability of error.

Note that if the capacities of all links in the network were greater than or equal to the source rate h, then the problem is trivial: every encoder in the network can simply replicate the original message X along each link. Each destination then receives the original message in its entirety on every input link. The problem becomes interesting when the link capacities are less than h, in which case the links can only carry partial information about X. The problem is then to coordinate a network-wide coding scheme such that every ultimate destination can still reconstruct X based on the multiple partial information it receives.

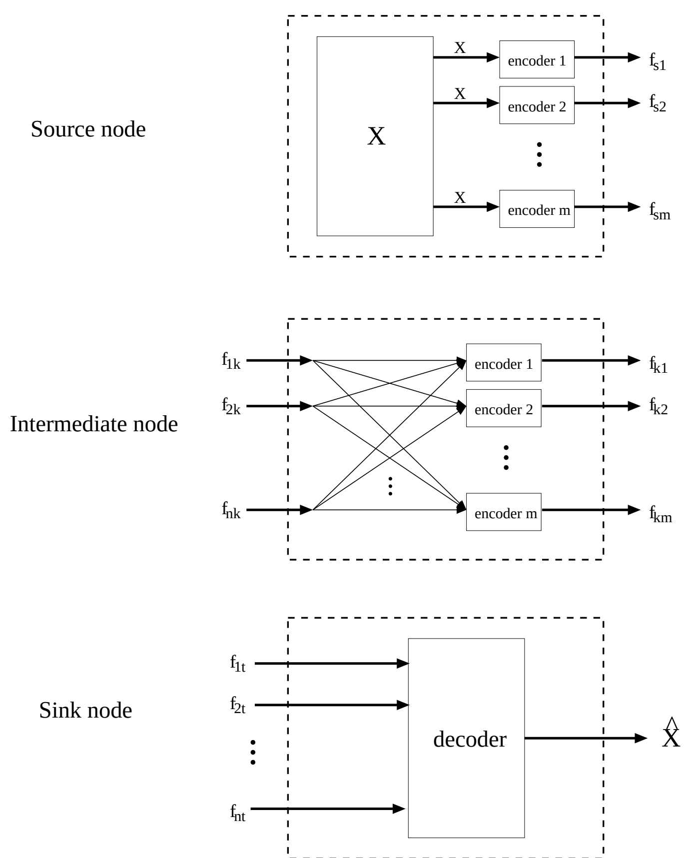

Figure 2 Diagram of nodes

The block-coding structure described above, where every encoder waits to collect all the information from the input links before generating a codeword, is referred to in the paper as a "β-code". Such codes make sense in a network without any cycles, since the information passing through such a network does so in a sequential manner. On the other hand, if a network contains cycles then it is possible that information sent from a node will cause future inputs to that node. Thus, each node will have to perform multiple encoding operations. To account for this, the authors introduce a more general class of codes called α-codes. These are more difficult to explain, however, and since most of the intuition in this paper can be obtained in acyclic networks, we will postpone discussion of this more general class of codes until later.

3 Preview of Results

The main contribution of this paper is a coding theorem for the single-source, multiple-destination case. The theorem states that a necessary and sufficient condition for each destination to be able to reconstruct the source is that the capacity of the minimum cut to every destination must be greater than or equal to the source rate. (For a review of minimum cuts and basic graph theory, refer to the Appendix).

For the general problem with multiple sources, the authors illustrate how this can be more difficult than the single-source problem. In particular, they give an example showing that a simple superposition of single-source codes is not always optimal.

3.1 Examples

We give two examples which illustrate the main result of this paper (shown in Figure 3.1). In both examples we assume that every link has unit capacity. In the first example, we can see by inspection that the capacity of the minimum cut to both destinations is 2 bits/sec. To achieve this, the source node can transmit 2 distinct bits per unit time along each output link, while the middle link $(3, 4)$ performs a mod-2 addition of the two bits. Thus, both destinations can reconstruct the original two bits every time unit.

The second example shows that bit-by-bit operations are not always optimal, and that block codes are sometimes necessary. The minimum cut to each of the six destinations is 2 bits/sec. Assume that the source transmits symbols b_1 and b_2 (which may not be binary in this case) along its two output links. Notice that the four middle links $(3, 7), (4, 8), (5, 9), (6, 10)$ each have access to both source symbols b_1, b_2. Let c_1, c_2, c_3, and c_4 denote the symbols transmitted along each of these four middle links, respectively. Each of the six destinations has access to a different pair of these four middle links. So another way to state the network coding problem is: we wish to encode two source symbols, (b_1, b_2), into four code symbols, (c_1, c_2, c_3, c_4), such that any two of the four code symbols can be used to recover (b_1, b_2). This is a classical problem in linear coding theory, and can be solved by Maximum-Distance-Separable (MDS) codes, such as the Reed-Solomon (RS) codes. Specifically, a (n, k) RS code encodes k "source" symbols from a field $GF(2^m)$ into n "code" symbols (from the same field) in such a way that any combination of k of the n code symbols is sufficient to recover the original k source symbols. An important condition for such a code to exist is that the field must have a sufficient number of elements: $2^m \geq n$. So in this example, where $n = 4$ and $k = 2$, we need $m \geq 2$ (a field of at least 4 elements).

This implies that the example cannot be solved with binary symbols. To see this, notice that it is not possible to find four bits to send along the middle links which allow all six destinations to recover both b_1 and b_2 every time unit (in the situation depicted in the diagram, the destinations t_3, t_5, t_6 can only decode at most one of the two source bits per unit time). Suppose, however, that we choose b_1 and b_2 to be symbols from $GF(4) = \{0, 1, x, 1 + x\}$. Then by assigning the four middle links to carry the symbols $(0, 1, b_1 + b_2, b_1 + xb_2)$ (where addition is mod-2 and multiplication is with respect to the irreducible polynomial $1 + x + x^2$), each of the six destinations can recover both b_1 and b_2 by simple operations on the received symbols.

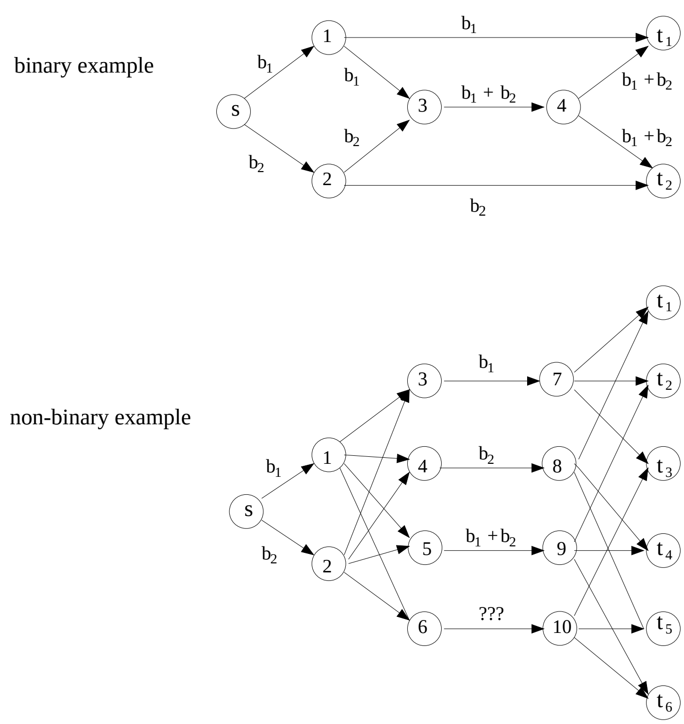

Figure 3.1 Examples of network coding

The decoding at each destination goes as follows: decoding for node t_1 is trivial since it receives both source symbols in their entirety. Nodes t_2 and t_4 each receive one of the source symbols as well as the mod-2 sum $b_1 + b_2$, hence can recover both source symbols by mod-2 addition. Node t_3 receives b_1 and the sum $b_1 + xb_2$. By mod-2 addition, it can create the pair (b_1, xb_2), which uniquely identify the two source symbols. For t_5, multiplying the received symbol b_2 by x and adding it to the other received symbol yields the pair (b_1, b_2). For t_6, mod-2 addition of the two received symbols yields $(x+1)b_2$, which uniquely identifies b_2, and this can be used to recover b_1 from $b_1 + b_2$. Assuming that the links are binary links, this code can be implemented by representing the elements from $GF(4)$ as pairs of bits. Each destination must wait two time units to collect a pair of bits from each input link before decoding the original symbols. As a result, the received rate is four bits per two seconds, or two bits/sec, corresponding to the capacity of the min-cut.

4 Related Work

4.1 Multilevel Diversity Coding

Multilevel diversity coding (MDC), first introduced by Roche [4] and Yeung [7], is a class of problems in multiterminal source coding. A MDC system consists of an information source, a set of encoders, and a set of decoders. Each encoder has access to the same information source, and each decoder has access to a particular subset of encoders (the particular assignment of subsets of encoders to decoders is called the *configuration* of the system). Decoders which have access to the same encoder see the same output from that encoder. The reconstruction requirements of each decoder may all be different. In particular, the set of decoders is partitioned into multiple *levels*, where decoders within the same level are subject to the same reconstruction requirements. The problem is to determine the coding rate region for a given configuration of the MDC system subject to certain distortion criteria.

The authors demonstrate that the MDC problem can be modeled as a network flow problem by appropriately choosing link capacities. An example is shown in Figure 4.1. In the corresponding network flow problem, every output link from the source and every input link into each destination have unbounded capacity. Hence, it is optimal for the source node to simply replicate the input source message along each of its outgoing links to nodes 1,2, and 3. This corresponds to the MDC scenario in which the 3 encoders have full access to the sources X_1, X_2. Similarly, nodes $4, 5$ and 6 can simply replicate the information that they receive from their input links. This corresponds to the 3 decoders in the MDC system having full access to the outputs of the encoders.

So the only coding that needs to be done in this network flow problem is in the three middle links, $(1, 4)$, $(2, 5), (3, 6)$ (these links correspond to the 3 encoders in the MDC system). The extra stage from nodes $1, 2, 3$ to nodes $4, 5, 6$ in the network flow problem is needed to make the network flow problem consistent with the MDC requirement that all decoders connected to the same encoder have access to the same information. The rate of the three middle links corresponds to the rate of the three encoders in the MDC problem. The problem is then to find the set of admissible rates (capacities) for the three middle links such that every destination can recover its desired messages.

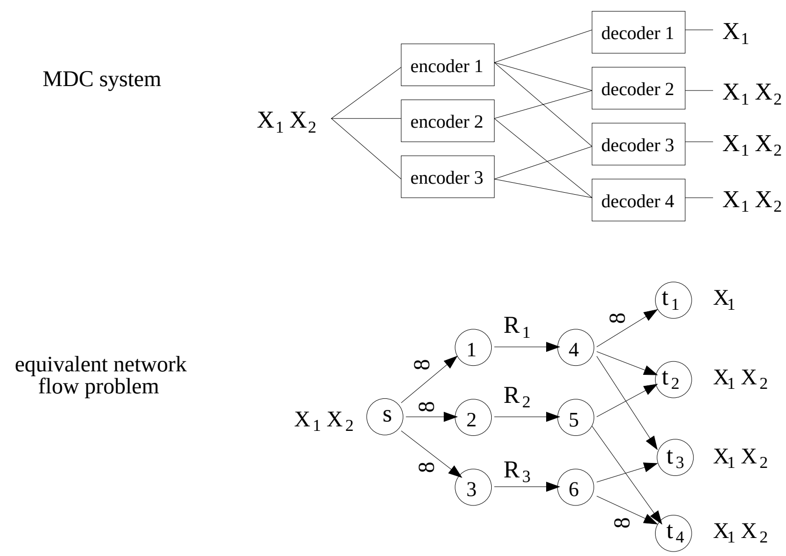

Figure 4.1 Example of an MDC system

The case of an MDC system with 2 decoding levels and arbitrary connectivity from source to destinations was solved by Yeung [7], and the case of 3 decoding levels with symmetric connectivity was solved by Roche, Yeung, and Hau [5].

The network information flow model presented in the current paper pedagogically includes the class of all MDC systems. The actual results obtained for the single-source network problem, however, only apply to the case of a MDC system with a single decoding level, hence adds nothing new to the results of [7].

4.2 Generalized Slepian-Wolf Coding

In [3], T.S. Han considered the problem of transmitting multiple correlated information sources to a single destination over a general network of discrete (noisy or noiseless) independent channels. This can be seen a generalized version of the classical Slepian-Wolf multiterminal source-coding problem, in that the decoders have indirect access to the encoders via intermediate relay nodes. The setup in [3] is almost identical to the current paper's: the intermediate nodes are allowed to decode and re-encode information, and the ultimate reconstruction assumes no distortion. The only differences are that [3] assumes multi-source/single-destination communication over possibly *noisy* channels (as opposed to the single-source/multi-destination communication over noiseless channels in the current paper). The main result in [3] gives a necessary and sufficient condition for a given network to be achievable, in the sense that all the sources can be reconstructed at the destination with arbitrarily small probability of error.

Interestingly, the authors of the current paper do not mention this reference, despite the fact that they make a connection between the two problems: the authors claim (without proof) that a scenario of multiple independent sources sending information to a common set of destinations can be modeled as a single-source problem by adding a fictitious source node connecting to all the original source nodes, where the capacity of the link connecting the fictitious node to source node i is set to the source rate h_i. Assuming this claim to be true, then the problem in [3] of multiple sources transmitting to a single destination over *noiseless* links can be solved by the results of the current paper.

Other references to classical multiterminal source coding can be found in Slepian and Wolf [6], Cover [1], and Cover and Thomas [2].

5 Main Theorem

We now present the main theorem of the current paper. Assume the source rate is h bits/sec. Recall that the encoder for link (i,j) selects a symbol from the alphabet $\{1,\ldots,\eta_{ij}\}$ to encode over link (i,j).

Definition: The tuple $(\mathbf{R},h,G)$ is "α-admissible" if for any $\epsilon > 0$ and for sufficiently large n, there exists an $(n,\eta_{ij},h-\epsilon)$ α-code such that for all $(i,j)\in E$:

$$n^{-1}\log\eta_{ij} \le R_{ij}+\epsilon$$

Note that for any finite n, α-codes are allowed to violate the link capacities of the network. Only in the limit as $n\to\infty$ are these codes feasible. We point out that the presence of the positive ϵ in the above definition is crucial to the proof of achievability, so that this slightly awkward definition of admissibility is necessary in proving the results of this paper.

Theorem 5.1 *the tuple $(\mathbf{R},h,G)$ is "α-admissible" if and only if h does not exceed any maximum flow from s to $t_1,\ldots,t_L$.*

5.1 Proof of the Converse

Assume that for any $\epsilon > 0$ and for sufficiently large n, there exists an $(n,\eta_{ij},h-\epsilon)$ α-code on G such that for all $(i,j)\in E$:

$$n^{-1}\log\eta_{ij} \le R_{ij}+\epsilon$$

- Let $w_{t_l}(X)$ denote all the information that destination node t_l receives from its input links: $w_{t_l}(X)=\{f_{it_l}(X)\}_{i:(i,t_l)\in E}$.
- Let $f_B(X)$ denote all the information that is sent through the cut $(B,\overline{B})$, for any set B: $f_B(X)=\{f_{ij}(X)\}_{(i,j):i\in B,j\in\overline{B}}$.
- Note that $X\to f_B(X)\to w_{t_l}(X)$ forms a Markov chain (since all the information received at a destination must have been derived from the information passing through a cut). Using the Data Processing Inequality, $I(X;w_{t_l}(X))\le I(X;f_B(X))$. Hence:

$$\begin{aligned} H(X)-H(X\mid w_{t_l}) &= I(X;w_{t_l}(X))\le I(X;f_B(X)) \\ &\le H(f_B(X))=H(\{f_{ij}(X)\}_{(i,j):i\in B,j\in\overline{B}}) \end{aligned}$$

Hence, minimizing over all cuts $(B, \overline{B})$ yields the desired result:

$$H(X) \leq H(f_B(X)) \leq \min_B H(f_B(X))$$

The right-hand-side is the capacity of the minimum cutset between s and t, which equals the maximum flow from s to t by the Max-Flow, Min-Cut Theorem.

- More rigorously:

$$H(X) \leq H(\{f_{ij}\}_{(i,j)\in(B,\overline{B})}) \leq \sum_{(i,j)\in(B,\overline{B})} H(f_{ij}) \leq \sum_{(i,j)\in(B,\overline{B})} \log \eta_{ij}$$

$$\begin{aligned} h - \epsilon &\leq \frac{1}{n} H(X) \leq \frac{1}{n} \sum_{(i,j):i\in B, j\in\overline{B}} \log \eta_{ij} \\ &\leq \sum_{(i,j):i\in B, j\in\overline{B}} (R_{ij} + \epsilon) \leq \sum_{(i,j):i\in B, j\in\overline{B}} R_{ij} + E\epsilon \\ &\leq \min_B \sum_{(i,j):i\in B, j\in\overline{B}} R_{ij} + E\epsilon \end{aligned}$$

Letting $\epsilon \to 0$ proves the theorem.

5.2 Achievability for Acyclic Networks

Assume that the link capacities $\{R_{ij}\}$ are such that the values of the maximum flow from s to $t_1, \ldots, t_L$ are all greater than or equal to h. Assume for now that the network contains no cycles. The general idea of the achievability construction is to first construct an α-admissible random codebook with a slightly larger message set, which has probability of error asymptotically going to zero with block length n, and from this derive a deterministic codebook with zero error by throwing away those messages which cannot be distinguished. The steps of the random code construction are as follows:

- The message set Ω is expanded to a larger set Ω', where $|\Omega'| = C|\Omega|$, $C > 1$.
- For every message $x \in \Omega'$, the encoder for each link (i, j) randomly selects a symbol with uniform distribution from $\{1, \ldots, \eta_{ij}\}$ to send over the link.
- At destination t_l, an error occurs if and only if the total information it receives from all input links is the same for any two distinct messages.
- By standard union bound arguments, the authors show that the overall probability of error (that is, the probability that any of the L destinations is in error) is upper-bounded by:

$$\Pr(\text{error}) < LC2^{|V|}2^{-n\rho}$$

where L denotes the length of the longest path in the network, and ρ is some positive constant less than ϵ. Notice that this upper bound approaches zero as the block length n goes to infinity.

- So the average number of messages in the set Ω' which can be correctly decoded must be at least:

$$(1 - LC2^{|V|}2^{-n\rho})C2^{nh}$$

For large enough n, this is greater than 2^{nh}. Hence, for sufficiently large n, we can construct Ω by picking any 2^{nh} distinguishable messages from Ω', and this yields an error-free α-admissible code.

We next give a simple example of a random code construction in Figure 5.2. The original message set has 2 symbols, and we expand it into a message set of 3 symbols. For each message in this expanded message set, each link in the network randomly selects a symbol from $\{a, b, c, d\}$. One possible result of the random selection is shown in the figure.

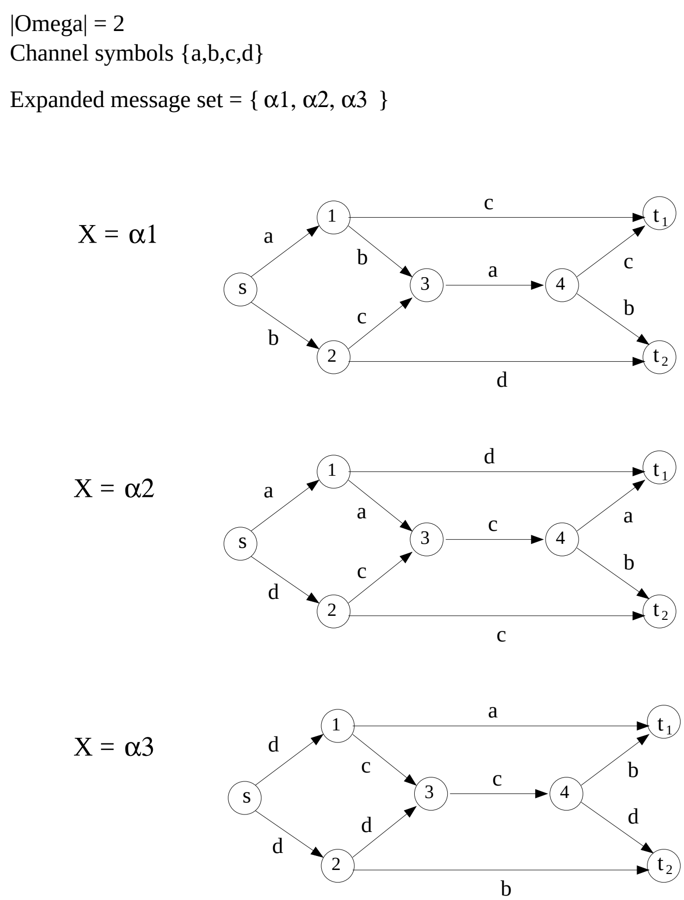

Figure 5.2 Example of a random code construction

At node 1, messages α_1 and α_2 are indistinguishable along input link $(s, 1)$. Hence, the output links from node 1 cannot distinguish between these two messages. Furthermore, node 3 cannot distinguish between the same two messages from either input link $(2, 3)$ or $(1, 3)$ (the latter because node 1 is itself confused). So the links $(1, t_1)$

and $(3, 4)$ are useless in this case, which means that destination node t_1 cannot distinguish between α_1 and α_2 (destination t_2, however, can recover the message via the information from input link $(2, t_2)$). Similarly, notice that messages α_2 and α_3 are indistinguishable at nodes 2 and 4. In this case, destination t_2 cannot distinguish these two messages. So clearly we cannot chose the pairs $\{\alpha_1, \alpha_2\}$ nor $\{\alpha_2, \alpha_3\}$ as our original message set. Fortunately, in this example the last pair of messages, α_1 and α_3, are distinguishable at every node. Therefore, if we choose the message set to be $\{\alpha_1, \alpha_3\}$, then the corresponding encodings shown in the figure yield an error-free deterministic network code.

5.3 Achievability for Networks with Cycles

For networks containing cycles, a slightly more elaborate coding method is used. First, the graph $G = (V, E)$ is used to create another graph $G^* = (V^*, E^*)$ containing no cycles, via the following method: let the set V^* consist of $\Lambda + 1$ layers of nodes, each of which is a copy of V: $V^* = \bigcup_{\lambda=0}^{\Lambda} V^{(\lambda)}$, where $V^{(\lambda)} = \{i^\lambda : i \in V\}$. We will see later that λ acts as a time parameter. The set of edges E^* consists of three types of edges:

1. $(s^{(0)}, s^{(\lambda)})$, $1 \leq \lambda \leq \Lambda$;
2. $(t_l^{(\lambda)}, t_l^{(\lambda)})$, $1 \leq \lambda \leq \Lambda - 1$;
3. $(i^{(\lambda)}, j^{(\lambda+1)})$, $(i, j) \in E$, $0 \leq \lambda \leq \Lambda - 1$.

In this new graph G^*, let s^* be the source and $t_l^* = t_l^\lambda$ be a sink corresponding to the sink t_l in G. Clearly, G^* is acyclic by construction.

The capacities of edges in G^* are defined by $R^*_{uv} = R_{ij}$ if $(u, v) = (i^\lambda, j^{\lambda+1})$ for some $(i, j) \in E$ and $1 \leq \lambda \leq \Lambda$. For all other links in G^*, set $R^*_{uv} = \infty$. The proof of achievability first constructs a β-code on this acyclic graph, and then proves that this code can be adapted to the original graph G. We outline the main points of the proof:

- The authors prove a lemma stating that if the value of a max-flow from s to t_l in G is greater than or equal to h, then the value of a max-flow from s^* to t_l^* in G^* is greater than or equal to $(\Lambda - d_l)h$, where d_l is the maximum length of a simple path (a path containing no cycles) from s to t_l.

- Using this lemma and the achievabilty result for acyclic networks in the previous section, we can conclude that there exists, for sufficiently large ν, a $(\nu, \eta^*_{uv}, (\Lambda - d)h)$ β-code which is α-admissible, where d is the longest path of any source-destination pair.

- Now letting $n = \Lambda\nu$, use this β-code on G^* to construct an $(n, \eta_{ij}, (\Lambda - d)h/\Lambda)$ α-code on G, where the channel symbol set is now the product of all the channel symbol sets from the Λ coding phases in G^*:

$$\eta_{ij} = \prod_{\lambda=0}^{\Lambda-1} \eta^*_{i^{(\lambda)} j^{(\lambda)}}$$

The coding session of this α-code consists of $\Lambda+1$ phases where in each phase λ, node i sends the information which its corresponding duplicate node sent in level λ of G^*.

- In phase $\Lambda+1$, each sink t_l uses the total received information to decode X.
- The rate on each link satisfies:

$$\begin{aligned}\frac{1}{n}\log \eta_{ij} &= (\Lambda\nu)^{-1}\log\prod_{\lambda=0}^{\Lambda-1}\eta^*_{i^{(\lambda)}j^{(\lambda+1)}} \\ &\leq (\Lambda)^{-1}\sum_{\lambda=0}^{\Lambda-1}(R^*_{i^{(\lambda)}j^{(\lambda+1)}}+\epsilon) \\ &= \Lambda^{-1}\sum_{\lambda=0}^{\Lambda-1}(R^*_{ij}+\epsilon) \\ &= R_{ij}+\epsilon\end{aligned}$$

- Finally, for any $\epsilon>0$, and for large enough Λ,

$$\frac{(\Lambda-d)h}{\Lambda} > h-\epsilon$$

- Therefore, we conclude that the network is α-admissible.

6 Multiple sources

Now suppose that there are multiple sources attempting to multicast information to possibly different sets of destinations. One possible coding scheme would be to code for each source independently and simply run all the single-source codes in parallel. The authors give an example showing that this is not always optimal, and that joint encoding of sources may be necessary. The example is shown in Figure 4.1, which was the MDC system. Given the equivalence between MDC and network flow problems, we will describe this example as an MDC problem, which was its original context. Assume that the source rate is 1 bit/sec. Suppose that we jointly encode the two sources at each encoder. One possibility is for encoder 1 to output X_1, encoder 2 to output X_2 and encoder 3 output the mod-2 sum (X_1+X_2). Then each decoder can recover its intended information and the resulting rate for each encoder is 1 bit/sec.

Now consider separately encoding the two sources. That is, each encoder has two sub-encoders, one for each source, running in parallel. The total encoding rate at encoder i is the sum of the rates for the individual sub-encoders: $R_i = r_{i1}+r_{i2}$. We will show that such a coding scheme cannot achieve the rate tuple $(1,1,1)$. Notice that since decoder 1 only has access to encoder 1, we must have $r_{11}=1$. In order to achieve $R_1=1$, this implies that $r_{12}=0$ (that is, decoder 1 ignores X_2), which in turn implies that decoder 3 must get X_2 completely from encoder 3, so that $r_{32}=1$.

To achieve $R_3 = 1$, this implies $r_{31} = 0$. Similarly, since decoder 2 must get X_2 from encoder 2 alone, $r_{22} = 1$, which implies $r_{21} = 0$ (to achieve $R_2 = 1$). At this point, we see that decoder 4 cannot receive X_1 since both $r_{21} = 0$ and $r_{31} = 0$.

A Max-flow Min-cut Theorem

Let $G = (V, E)$ be a directed graph consisting of: set of nodes V interconnected by a set of edges E. A *network* $N = (s, t, V, E, \mathbf{R})$ is a graph (V, E) together with a source s, a destination t, and a capacity R_{ij} for each edge $(i, j) \in E$.

A *flow* (or $s - t$ flow) $\mathbf{F}$ in N is a vector in $\mathbf{R}^{|E|}$ (one component $F(i, j)$ for each edge $(i, j) \in E$) such that:

1. $0 \leq F(i, j) \leq R_{ij}$ for all $(i, j) \in E$.
2. $\sum_{i:(i,j)\in E} F(i, j) = \sum_{k:(j,k)\in E} F(j, k)$ for all $j \in V - \{s, t\}$.

The *value* of an $s - t$ flow $\mathbf{F}$ is $\sum_{k:(s,k)} F(s, k)$. The *maximum flow* from s to t is the $s - t$ flow with maximum value.

An $s - t$ *cut* is a partition $(B, \overline{B})$ of nodes V into sets B and $\overline{B} = V - B$ such that $s \in B$ and $t \in \overline{B}$. The *capacity* of an $s - t$ cut is

$$C(B, \overline{B}) = \sum_{(i,j)\in E: i\in B, j\in \overline{B}} R_{i,j} \tag{1}$$

Theorem A.1 (Max-flow, min-cut) *The value v of any $s - t$ flow is no greater than the capacity $C(B, \overline{B})$ of any $s - t$ cut. Furthermore, the value of the maximum flow equals the capacity of the minimum cut.*

When R_{ij} are all rational, the maximum flow vector can be explicitly constructed using the Ford-Fulkerson algorithm.

For a single source and single destination, this max-flow vector consists of a set of disjoint paths, so encoding is trivial: send a distinct bit stream along each path.

For a single source multicasting to multiple destinations, the max-flow vector to each destination may not be simultaneously achievable, hence the encoding strategy becomes more difficult (possibly need to re-encode data at intermediate nodes).

References

[1] T. Cover. A proof of the data compression theorem of slepian and wolf for ergodic sources. *IEEE Trans. Info. Theory*, IT-21:226–228, 1975.

[2] T. Cover and J. Thomas. *Elements of Information Theory*, 1991.

[3] T.S. Han. Slepian-wolf-cover theorem for networks of channels. *Information and Control*, 47:67–83, 1980.

[4] J. Roche. Distributed information storage. *Ph.D. dissertation, Stanford University*, March 1992.

[5] J. Roche, R. Yeung, and K. Hau. Symmetrical multilevel diversity coding. *IEEE Trans. Info. Theory*, IT-43:1059–1064, May 1997.

[6] D. Slepian and J.K. Wolf. Noiseless coding of correlated information sources. *IEEE Trans. Info. Theory*, IT-19:471–480, 1973.

[7] R. Yeung. Multilevel diversity coding with distortion. *IEEE Trans. Info. Theory*, IT-41:412–422, March 1995.