

Lecture 8

1 Linear Information Flow Algorithm

We now present a centralized algorithm for multicast network code construction. We assume a directed acyclic graph $G = (V, E)$ with source node S having h data sources and a set of terminals \mathcal{T} . We assume that the max-flow(S, T_i) is always greater than or equal to h for all $T_i \in \mathcal{T}$.

1.1 Preprocessing Step:

1. Find h edge-disjoint paths from S to T_i for all $T_i \in \mathcal{T}$, and denote this as the set $\mathcal{P}_i = \{P_{i,j}\}_{j=1}^h$, where $P_{i,j}$ is the j th path in the set.
2. Form a new graph G' consisting of all of these paths where the set of edges E' in G' is equal to all edges in $\cup_{i=1}^{|\mathcal{T}|} \mathcal{P}_i$.
3. Number the edges in G' such that if there exists a path from e_i to e_j , then $i < j$.

We now define several notions we will use subsequently.

Definition 1. *Coding point.* Any edge in G' that has two or more incoming edges connecting to it.

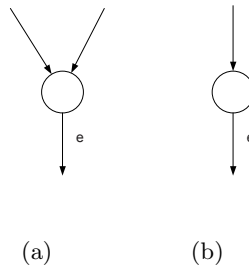


Figure 1: (a) e is not a coding point. (b) e is not a coding point.

For example, in Figure 1(a), e is a coding point; in Figure 1(b), e is not a coding point. If in G' , δ is a coding point, we define $T(\delta)$ to be the set of terminals whose path set contains δ . For $T_i \in T(\delta)$, the coding point δ appears at most in one path in \mathcal{P}_i , since the paths in \mathcal{P}_i are disjoint. We can further define $\text{prev}^i(\delta)$ as the predecessor coding point to δ along the path in \mathcal{P}_i , and because the paths are all disjoint in \mathcal{P}_i , $\text{prev}^i(\delta)$ is unique. Let $\beta(\delta)$ denote the global coding vector of δ . The parents of δ are defined as the set of predecessor coding points of δ that lie on the path from S to $T_i, \forall T_i \in T(\delta)$. The number of the parents of δ is at most the number of terminals.

Definition 2. *Frontier edge set F_i^t .* The frontier edge set for terminal T_i at time t , F_i^t is a set of h coding points, $e_1^{(i)}, \dots, e_h^{(i)}$ such that $e_j^{(i)} \in P_{ij}$ and $e_j^{(i)} < e_t$. It is initialized at $t = 0$ to be $\{e_{-1}, e_{-2}, \dots, e_{-h}\}$ for all T_i .

The frontier edge set keeps track of the most recently visited coding point δ in each of the h disjoint paths in \mathcal{P}_i .

Definition 3. *Frontier edge matrix M_i^t . The frontier edge matrix for T_i at time t is an $h \times h$ matrix over $GF(q)$ denoted by M_i^t whose rows are the global coding vectors of the coding points in F_i^t . The frontier edge matrices are initialized to be $h \times h$ identity matrix for all terminals.*

When processing coding point δ , the coding vectors of all parents of δ belong to the respective frontier edge matrices. The LIF algorithm maintains the frontier edge sets and frontiers edge matrices as explained below.

At step k , the algorithm considers edge $e_k \in E'$. If e_k is not a coding point, preserve F_i^k and M_i^k ; if it is a coding point, then denote it as δ_k and do the following.

1. Identify the set $T(\delta_k)$.
2. The new frontier edge sets F_i^k and frontier edge matrices M_i^k are decided in the following way:
 - F_i^k is equal to F_i^{k-1} except that $\text{prev}^i(\delta_k)$ in F_i^{k-1} is replaced by δ_k in F_i^k , $\forall T_i \in T(\delta_k)$. For $T_i \notin T(\delta_k)$, we set $F_i^k = F_i^{k-1}$.
 - We find a coding vector $\beta(\delta_k)$, such that $\beta(\delta_k)$ belongs to the span of the parents' coding vectors, and $\dim(M_i^{k-1} \setminus \beta(\text{prev}^i(\delta_k)) \cap \beta(\delta_k)) = h$. M_i^k is equal to M_i^{k-1} except that $\beta(\text{prev}^i(\delta_k))$ in M_i^{k-1} is replaced by $\beta(\delta_k)$ in M_i^k , $\forall T_i \in T(\delta_k)$. $M_i^k = M_i^{k-1}$, $\forall T_i \notin T(\delta_k)$.

It remains to show that a suitable $\beta(\delta_k)$ can always be found.

Claim 1. *Suitable $\beta(\delta)$ can always be found over $GF(q)$ when q is large enough.*

Proof: Any coding point δ_k has at most $|T|$ parents. Suppose that δ_k has $k' \leq |T|$ parents. Let $\mathcal{L}\{v_1, v_2, \dots, v_{k'}\}$ represent the span of the vector $v_1, v_2, \dots, v_{k'}$. The dimension of the vector space of $v_1, v_2, \dots, v_{k'}$ is denoted as $\dim\{v_1, v_2, \dots, v_{k'}\} = \text{number of independent column vectors in the set}$. Suppose that $\dim\{v_1, v_2, \dots, v_{k'}\} = k \leq k'$. Then the total number of nonzero vectors in $\mathcal{L}\{v_1, v_2, \dots, v_{k'}\} = q^k - 1$.

We know that any linear combination of vectors in $\mathcal{L}\{v_1, v_2, \dots, v_{k'}\}$ that has a nonzero coefficient associated with v_i does not belong to $\mathcal{L}\{M_i - \{v_i\}\}$. This is because by assumption each M_i is full rank. This implies that $\{M_i - \{v_i\}\}$ has rank $h - 1$ and does not contain any linear combination that has a nonzero coefficient associated with v_i . Therefore, if a nonzero vector belongs to both $\mathcal{L}\{M_i - \{v_i\}\}$ and $\mathcal{L}\{v_1, v_2, \dots, v_{k'}\}$ must have a zero coefficient for v_i and the number of such vectors in $\mathcal{L}\{v_1, v_2, \dots, v_{k'}\}$ is at most $q^{k-1} - 1$. Thus, we have established that the number of nonzero vectors in $\mathcal{L}\{M_i - \{v_i\}\} \cap \mathcal{L}\{v_1, v_2, \dots, v_{k'}\} \leq q^{k-1} - 1$. Now the total number of vectors that belongs to one of these intersections $\mathcal{L}\{M_i - \{v_i\}\} \cap \mathcal{L}\{v_1, v_2, \dots, v_{k'}\}$ for $i = 1, \dots, k'$ is at most $k'(q^{k-1} - 1) < k'q^{k-1} < q^k$ if $k' < q$.

So we conclude that number of vectors that do not belong to any of these intersection is at least $q^k - k'q^{k-1}$ that is strictly positive for $k' < q$ which means that a vector that does not belong to any of these intersections $\mathcal{L}\{M_i - \{v_i\}\} \cap \mathcal{L}\{v_1, v_2, \dots, v_{k'}\}$ can be found.

In our problem, k' is the number of terminals. It means that if $q > |T|$, then we can find a suitable coding vector for all δ_k at step k . This implies that a multicast network code can be constructed using the LIF algorithm.

2 An Example of the LIF Algorithm

We now show an example of the LIF algorithm. Consider the network shown in Figure 2. Suppose that Figure 2 is the G' obtained from the preprocessing steps for LIF algorithm. There is one source S and two terminals T_1 and T_2 in the network. Source S produces two source messages and both terminals want to receive the two source messages. From the figure, we can see that the path sets from S to both terminals are as follows.

$$\mathcal{P}_1 = \{\{e_{-1}, e_0, e_2\}, \{e_{-2}, e_1, e_4, e_6, e_7\}\} \text{ and } \mathcal{P}_2 = \{\{e_{-1}, e_0, e_3, e_6, e_8\}, \{e_{-2}, e_1, e_5\}\}$$

Initially at time 0, both frontier edge sets F_i^0 for $i = 1, 2$ were initialized to be $\{e_{-1}, e_{-2}\}$. Both frontier edge matrices M_i^0 for $i = 1, 2$ were initialized to be the 2×2 identity matrix. The global coding vectors for source edges $\{e_{-1}, e_{-2}\}$ were initialized to be $[0, 1]$ and $[1, 0]$ respectively.

At time 0, we consider assigning a global coding vector for e_0 in the graph which should be from the span of the coding vectors of the incoming edges and at the same time make the matrix M_i invertible. Let us assign a global vector $[0 \ 1]$ for e_0 . Now we update all F_i for $i = 1, 2$ by substituting the predecessor e_{-1} by e_0 . At the same time, by examining the path set P_1 and P_2 , we see that e_0 exists in both edge sets P_1 and P_2 , so we need to update both M_1 and M_2 . The coding vector $\beta(e_t)$ enters the $M_i^{(1)}$ for both $i = 1, 2$. Note that here $\beta(e_t) = [1 \ 0]$ which belongs to the span of the removed row vectors. The condition that $M_1^{(1)}, M_2^{(1)}$, remain full rank holds here since they continue to be the identity matrices. The algorithm continues as explained in the previous section.

In Figure 2 the global vectors for all edges in our example are labeled besides corresponding edges. The update process for the frontier edge sets and the frontier edge matrices of the two terminals are shown in Figure 3.

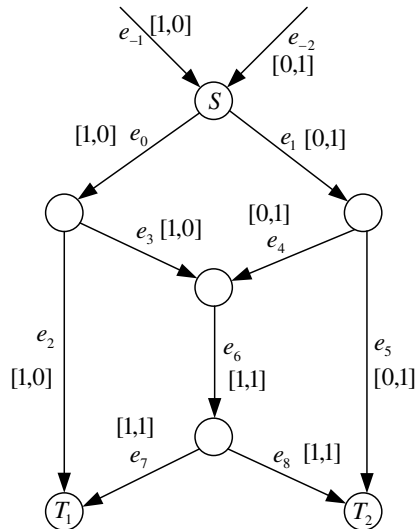


Figure 2: Example network in LIF algorithm

steps	F_1	F_2	M_1	M_2
0	$\{e_{-1}, e_{-2}\}$	$\{e_{-1}, e_{-2}\}$	$I_{2 \times 2}$	$I_{2 \times 2}$
1	$\{e_0, e_{-2}\}$	$\{e_0, e_{-2}\}$	$I_{2 \times 2}$	$I_{2 \times 2}$
2	$\{e_0, e_1\}$	$\{e_0, e_1\}$	$I_{2 \times 2}$	$I_{2 \times 2}$
3	$\{e_2, e_1\}$	$\{e_0, e_1\}$	$I_{2 \times 2}$	$I_{2 \times 2}$
4	$\{e_2, e_1\}$	$\{e_3, e_1\}$	$I_{2 \times 2}$	$I_{2 \times 2}$
5	$\{e_2, e_4\}$	$\{e_3, e_1\}$	$I_{2 \times 2}$	$I_{2 \times 2}$
6	$\{e_2, e_4\}$	$\{e_3, e_5\}$	$I_{2 \times 2}$	$I_{2 \times 2}$
7	$\{e_2, e_6\}$	$\{e_6, e_5\}$	$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$

Figure 3: Updating process for frontier edge sets and frontier edge matrix

3 Understanding some special network connections

Using the algebraic approach, the multicast problem is equivalent to finding a set of coefficients such that the determinants for multiple terminals are nonzero simultaneously. For a more general network problem, for example a unicast problem, we have to find a set of coefficients such that some determinants are nonzero and the interference matrices are zero matrices. This is a difficult problem in general and no systematic techniques are known for the general case. However, for some special network connections, useful inferences can be drawn.

3.1 Single source, multiple disjoint requests

In a directed acyclic graph $G = (V, E)$, where V denotes the set of vertices and E denotes the set of edges, let $S \in V$ denote the source, $\mathcal{T} = \{T_1, T_2, \dots, T_k\}$ denote the set of terminals where $T_i \in V$. Let $X(S)$ be the set of all messages at S , $X(S, T_i)$ be the set of messages requested by T_i , Z_{T_i} be the set of received messages at node T_i . We also have $X(S, T_i) \cap X(S, T_j) = \emptyset, \forall i \neq j$. The network is shown in figure 4(a).

Theorem 2. *The capacity region for single source multiple terminals multiple unicast problem is: the cut value $(S, T) \geq \sum_{T_i \in T} |X(S, T_i)|, \forall T \subset \mathcal{T}$.*

Proof: Consider the transfer matrix between S and $\{T_1, T_2, \dots, T_k\}$

$$[Z_{T_1}, \dots, Z_{T_k}] = [X(S, T_1)_1, \dots, X(S, T_1)_{|X(S, T_1)|}, \dots, X(S, T_k)_1, \dots, X(S, T_k)_{|X(S, T_k)|}]G$$

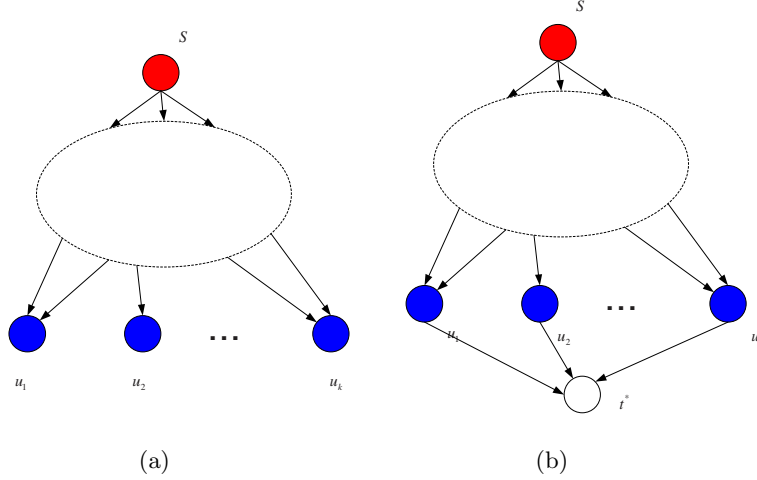


Figure 4: (a) The network $G = (V, E)$. (b) The auxiliary network G^* .

Where

$$G = \begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1k} \\ \vdots & \ddots & \vdots \\ \alpha_{k1} & \cdots & \alpha_{kk} \end{pmatrix}$$

in which α_{ij} is an $|X(S, T_i)| \times |X(S, T_j)|$ matrix. We hope that α_{ii} could be an identity matrix for $T_i \in \mathcal{T}$, and α_{ij} be zero matrix $\forall i \neq j$, which makes G a identity matrix with size $|X(S)| \times |X(S)|$. However, by the constraints of the network topology, we cannot find such a G . Because $\text{cut}(S, T) \geq \sum_{T_i \in T} |X(S, T_i)|, \forall T \subset \mathcal{T}$, we are able to find a G that is a full rank matrix by choosing the coefficients properly. Then we define a precoding matrix A , such that $A = G^{-1}$. At the source instead of transmitting $[X(S, T_1), X(S, T_2), \dots, X(S, T_k)]$, we transmit $[X(S, T_1), X(S, T_2), \dots, X(S, T_k)]A$. Then the terminals receive $[X(S, T_1), X(S, T_2), \dots, X(S, T_k)]AG = [X(S, T_1), X(S, T_2), \dots, X(S, T_k)]$, which means terminal T_i can recover $X(S, T_i)$ without interference. \square

It seems network coding is essential in proving the capacity region. However, we next show that routing is enough in proving the capacity region. Now we construct an auxiliary graph G^* in which we append a virtual super terminal t^* to G , and connect each terminal T_i to t^* . Figure 4(b) illustrates the construction. The capacity for edge (T_i, t^*) is set to be $|X(S, T_i)|, \forall T_i \in \mathcal{T}$.

Claim 3. *The max flow from S to t^* is $\sum |X(S, T_i)| = |X(S)|$.*

proof: We first find the minimum cut for the auxiliary graph G^* . Then by the minimum cut and maximum flow theorem, we have the maximum flow equal to $\sum |X(S, T_i)| = |X(S)|$

Case1 If a cut contains the source node S and all terminal nodes T_i , and does not contain t^* (the rest of the nodes can be either in the cut or not), then the cut value is least $\sum |X(S, T_i)| = |X(S)|$ (since the value from \mathcal{T} to t^* is $\sum |X(S, T_i)| = |X(S)|$).

Case2 Assume a set of terminals $T, T \subset \mathcal{T}$ and t^* are not in the cut. Then the cut value is at least $\text{cut}(S, T) + \sum_{T_i \in \mathcal{T} \setminus T} |X(S, T_i)|$. From the theorem we have $\text{cut}(S, T) \geq \sum_{T_i \in T} |X(S, T_i)|$, then the minimum cut in G^* is at least $\sum |X(S, T_i)| = |X(S)|$.

The minimum cut is equal to maximum flows. We can find $|X(S, T_i)|$ disjoint paths from S to t^* , and $|X(S, T_i)|$ of them have to go through $T_i, \forall i$. Then we can support multiple unicast using routing. \square