

Construction of Network Error Correction Codes in Packet Networks

Xuan Guang¹ Fang-Wei Fu¹ Zhen Zhang²

¹ Chern Institute of Mathematics
Nankai University

² Department of Electrical Engineering
University of Southern California

The 2011 International Symposium on Network Coding

Beijing, China

July 25-27, 2011

Outline

- 1 Background on Network Error Correction Coding
- 2 Linear Network Error Correction Codes (LNEC codes)
- 3 Constructive Proof of The Refined Singleton Bound on LNEC
- 4 Constructive Algorithm for General LNEC Codes

www.leaderstudio.net

Part I:

Background on Network Error Correction Coding

www.leaderstudio.net

Background

- Network coding has been extensively studied under the assumption that the channels of networks are error-free.
- Unfortunately, all kinds of errors may occur in practical network communications:
 - ① random errors;
 - ② erasure errors (packet losses);
 - ③ error in header;
 - ④ malicious attack;
 - ⑤

In order to deal with such problems efficiently, network error correction coding (NEC) was proposed by Cai and Yeung in 2002.

Background

- Because of its potential use in network communications, NEC has attracted a lot of attention and developed rapidly.
- For NEC Theory, there are two main distinct research approaches:
 - ① represent the messages by sequences as classical method, such as [Cai and Yeung Parts I, II 2006], [Zhang 2008], [Yang *et al.* 2011], etc;
 - ② represent the messages by subspaces of a fixed linear space as a totally new method, such as [Koetter and Kschischang 2008], [Silva, Kschischang, and Koetter, 2008], etc.

www.leaderstudy.net

Part II:

Linear Network Error Correction Codes

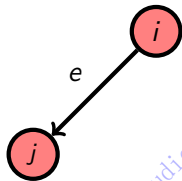
www.leaderstudio.net

Basic Definitions and Notation

- Let $G = (V, E)$ be a single source multicast (one-to-many) network.
- A direct edge $e = (i, j) \in E$ represents a channel leading from node i to node j .
- Node i is called the tail of e and node j is called the head of e , respectively written as:

$$i = \text{tail}(e), j = \text{head}(e).$$

- Correspondingly, the channel e is called an outgoing channel of i and an incoming channel of j .



www.leaderstudio.net

Basic Definitions and Notation

- For a node $i \in V$, define the following two sets:

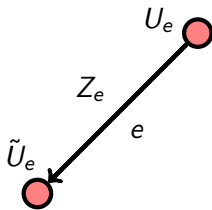
$$Out(i) = \{e \in E : tail(e) = i\},$$

$$In(i) = \{e \in E : head(e) = i\}.$$

- The source node s has no incoming channels, each sink node has no outgoing channels. But we use the concept of imaginary incoming channels of the source node s and assume that these imaginary incoming channels provide the source messages to s .

Linear Network Error Correction Codes

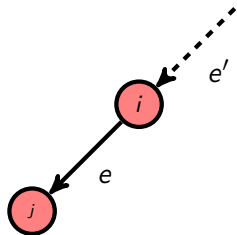
- If there is an error on channel e , the output of the channel is $\tilde{U}_e = U_e + Z_e$, where U_e is the message which should be transmitted over the channel e and Z_e is the error occurred on e .



www.leaderstudio.net

Linear Network Error Correction Codes

- We treat Z_e as a message called error message. So for the channel e , introduce the imaginary channel e' , and assume that Z_e is injected into the channel e by e' .



For each channel $e \in E$, an imaginary channel e' is introduced. The network with imaginary channels is called the **extended network** and denoted by $\tilde{G} = (\tilde{V}, \tilde{E})$.

Linear Network Error Correction Codes

For \tilde{G} , the global encoding kernel \tilde{f}_e for each $e \in \tilde{E}$ is an $(\omega + |E|)$ -dimensional column vector and the entries can be indexed by the channels in $\text{In}(s) \cup E$.

- 1 For **imaginary message channels** d'_i ($1 \leq i \leq \omega$) and **imaginary error channels** $e' \in E'$, define

$$\tilde{f}_{d'_i} = 1_{d'_i} \text{ and } \tilde{f}_{e'} = 1_{e'},$$

where 1_d is the indicator function of $d \in \text{In}(s) \cup E$.

- 2 For other global encoding kernels $\tilde{f}_e, e \in E$:

$$\tilde{f}_e = \sum_{d \in \text{In}(\text{tail}(e))} k_{d,e} \tilde{f}_d + 1_e.$$

www.leaderstudio.net

Linear Network Error Correction Codes

The matrix $\tilde{F}_t = [\tilde{f}_e : e \in \text{In}(t)]$ is called the decoding matrix at sink node t , and use $\text{row}_t(d)$ to denote the row vector of \tilde{F}_t corresponding to the channel d , $d \in \text{In}(s) \cup E$.

Definition

Let ρ be an error pattern, define

$$\Delta(t, \rho) = \langle \{\text{row}_t(d) : d \in \rho\} \rangle;$$

$$\Phi(t) = \langle \{\text{row}_t(d) : d \in \text{In}(s)\} \rangle;$$

where we call $\Delta(t, \rho)$ the error space of error pattern ρ and $\Phi(t)$ the message space.

Linear Network Error Correction Codes

Definition

- A linear network error correction code is called a regular code if for any $t \in \mathcal{T}$, $\dim(\Phi(t)) = \omega$.
- The minimum distance of a regular network error correction code at a sink node t is defined by

$$d_{\min}^{(t)} = \min\{|\rho| : \dim(\Delta(t, \rho) \cap \Phi(t)) > 0\}.$$

www.leaderstudio.net

Part III:

Constructive Proof of The Refined Singleton Bound on LNEC

www.leaderstudio.net

The Refined Singleton Bound

Theorem (The Refined Singleton Bound)

Let $d_{\min}^{(t)}$ be the minimum distance of a regular linear network error correction code at a sink node $t \in T$. Then

$$d_{\min}^{(t)} \leq \delta_t + 1,$$

where $\delta_t = C_t - \omega$ is the redundancy of the sink node t .

www.leaderstudio.net

The Refined Singleton Bound

Theorem (The Refined Singleton Bound)

Let $d_{\min}^{(t)}$ be the minimum distance of a regular linear network error correction code at a sink node $t \in T$. Then

$$d_{\min}^{(t)} \leq \delta_t + 1,$$

where $\delta_t = C_t - \omega$ is the redundancy of the sink node t .

Remark:

- This refined Singleton bound is achievable.

The Refined Singleton Bound

Theorem (The Refined Singleton Bound)

Let $d_{\min}^{(t)}$ be the minimum distance of a regular linear network error correction code at a sink node $t \in T$. Then

$$d_{\min}^{(t)} \leq \delta_t + 1,$$

where $\delta_t = C_t - \omega$ is the redundancy of the sink node t .

Remark:

- This refined Singleton bound is achievable.
- A linear network error correction code is called network error correction maximum distance separable (MDS) code, or network MDS code for short, if it satisfies this bound with equality.

Theorem

If $|\mathcal{F}| \geq \sum_{t \in T} |R_t(\delta_t)|$, then there exist linear network error correction MDS codes, i.e., for all $t \in T$,

$$d_{\min}^{(t)} = \delta_t + 1,$$

where

$$R_t(\delta_t) = \{\text{error pattern } \rho : |\rho| = \text{rank}_t(\rho) = \delta_t\}.$$

www.leaderstudio.net

Idea of Proof — Preparation:

- 1 Let $G = (V, E)$ be an one-to-many network and $\tilde{G} = (\tilde{V}, \tilde{E})$ be the extended network of G . Give an upstream-to-downstream order on the channel set E which is consistent with the partial order of all channels.

www.leaderstudio.net

Idea of Proof — Preparation:

- 1 Let $G = (V, E)$ be an one-to-many network and $\tilde{G} = (\tilde{V}, \tilde{E})$ be the extended network of G . Give an upstream-to-downstream order on the channel set E which is consistent with the partial order of all channels.
- 2 For each $t \in T$ and each $\rho \in R_t(\delta_t)$, there exist $(\omega + \delta_t)$ channel disjoint paths from either $In(s) = \{d'_1, d'_2, \dots, d'_\omega\}$ or $\rho' = \{e' : e \in \rho\}$ to t , and the $(\omega + \delta_t)$ paths satisfy the properties that
 - there are exactly δ_t paths from ρ' to t , and ω paths from $In(s)$ to t ;
 - these δ_t paths from ρ' to t start with the distinct channels in ρ' and for each path, if it starts with $e' \in \rho'$, then it passes through $e \in \rho$. (Refer to Lemma 2 and Corollary 3 in paper.)

And $\mathcal{P}_{t,\rho}$ denotes the set of these $\omega + \delta_t = C_t$ channel disjoint paths.

Idea of Proof — Main Idea:

Updating all channels according to the upstream-to-downstream order,

- define a dynamic channel set $CUT_{t,\rho}$ for each sink node $t \in T$ and each $\rho \in R_t(\delta_t)$, in which the elements are the latest $\omega + \delta_t$ channels on all paths in $\mathcal{P}_{t,\rho}$;
- choose the proper extended global encoding kernel for each $e \in E$ such that, at each moment, for all sink node $t \in T$ and all $\rho \in R_t(\delta_t)$, all matrices $\left[\tilde{f}_e^\rho : e \in CUT_{t,\rho} \right]$ are full rank, where \tilde{f}_e^ρ is an $(\omega + |\rho|)$ -dimensional column vector obtained from $\tilde{f}_e = (\tilde{f}_e(d) : d \in In(s) \cup E)$ by removing all entries $\tilde{f}_e(d)$ for $d \notin In(s) \cup \rho$.

Idea of Proof — Main Idea:

Mathematically, assume that the channel $e \in Out(i)$ will be updated,

- if $e \notin \cup_{t \in T} \cup_{\rho \in R_t(\delta_t)} E_{t,\rho}$, choose $\tilde{f}_e = 1_e$;
- otherwise, choose

$$\tilde{g}_e \in \tilde{\mathcal{L}}(\ln(i) \cup \{e'\}) \setminus \cup_{t \in T} \cup_{\substack{\rho \in R_t(\delta_t) \\ e \in E_{t,\rho}}} [\mathcal{L}^\rho(CUT_{t,\rho} \setminus \{e(t,\rho)\}) + \mathcal{L}^{\rho^c}(\ln(i) \cup \{e'\})], \quad (1)$$

where, for any channel set B , $\tilde{\mathcal{L}}(B) = \langle \{\tilde{f}_e : e \in B\} \rangle$,

$\mathcal{L}^\rho(B) = \langle \{f_e^\rho : e \in B\} \rangle$, $\mathcal{L}^{\rho^c}(B) = \langle \{f_e^{\rho^c} : e \in B\} \rangle$, and $e(t,\rho)$

denotes the previous channel of e on the path in $\mathcal{P}_{t,\rho}$ which e locates on. Further, let

$$\tilde{f}_e = \begin{cases} \tilde{g}_e + 1_e & \text{if } \tilde{g}_e(e) = 0, \\ \tilde{g}_e(e)^{-1} \cdot \tilde{g}_e & \text{otherwise.} \end{cases}$$

Idea of Proof — Final Proof:

Updating all channels in E by the same method, all $\tilde{f}_e, e \in E$ are well-defined. And we can prove the following two conclusions to complete the proof.

- For each $t \in T$, $d_{\min}^{(t)} = \delta_t + 1$.
- There exists column vector \tilde{g}_e satisfying (1), if the base field size is greater than or equal to $\sum_{t \in T} |R_t(\delta_t)|$.



www.leaderstudio.net

The Required Field Size

The required field size for the existence of network MDS codes can be smaller further than previous known results, even much smaller in some cases.

Corollary

For any one-to-many network, we have

$$\sum_{t \in T} |R_t(\delta_t)| < \sum_{t \in T} \binom{|E|}{\delta_t}.$$

www.leaderstudio.net

Example

Let G be a **combination network** with $N = 6$ and $k = 4$, where N is the number of internal nodes with one and only one channel from the source node s to each internal node and arbitrary k internal nodes are connective with one and only one sink node. For the network G , let $\omega = 2$, and thus $\delta_t = 2$ for each sink $t \in T$, we have

- $\sum_{t \in T} |R_t(\delta_t)| = 15 \times 24 = 360$,
- $\sum_{t \in T} \binom{|E|}{\delta_t} = 15 \times \binom{66}{2} = 32175$.

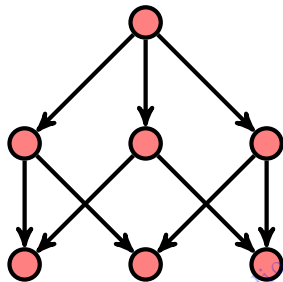


Figure: Combination Network with $N = 3, k = 2$.

Part IV:

Constructive Algorithm for General LNEC codes

www.leaderstudio.net

Constructive Algorithm

- The constructive proof motivates us to propose an algorithm for designing network MDS codes.
- Actually, we can also use this algorithm to construct general linear network error correction codes with some certain error correction capabilities so long as to change some parameters.

www.leaderstudio.net

Constructive Algorithm

Algorithm

Input: The single source multicast network $G = (V, E)$, the information rate $\omega \leq \min_{t \in T} C_t$, and the nonnegative integers $\beta_t \leq \delta_t$ for each $t \in T$.

Output: Extended global kernels (forming a linear network error correction code).

Initialization:

- 1 For each $t \in T$ and each $\rho \in R_t(\beta_t)$, find $(\omega + \beta_t)$ channel disjoint paths $\mathcal{P}_{t,\rho}$ from $In(s)$ or ρ' to t satisfying Preparation 2,
- 2 For each $t \in T$ and each $\rho \in R_t(\beta_t)$, initialize dynamic channel sets $CUT_{t,\rho} = In(s) \cup \rho' = \{d'_1, d'_2, \dots, d'_w\} \cup \{e' : e \in \rho\}$, and the extended global encoding kernels $\tilde{f}_e = 1_e$ for all imaginary channels $e \in In(s) \cup E'$.

for each channel $e \in E$ **do**

if $e \notin \cup_{t \in T} \cup_{\rho \in R_t(\beta_t)} E_{t,\rho}$ **then**

$\tilde{f}_e = 1_e$, all $CUT_{t,\rho}$ remain unchanged.

else if $e \in \cup_{t \in T} \cup_{\rho \in R_t(\beta_t)} E_{t,\rho}$ **then**

$\tilde{g}_e \in \tilde{\mathcal{L}}(\ln(i) \cup \{e'\}) \setminus \cup_{\substack{t \in T \\ e \in E_{t,\rho}}} \cup_{\rho \in R_t(\beta_t)} [\mathcal{L}^\rho(CUT_{t,\rho} \setminus \{e(t,\rho)\}) + \mathcal{L}^{\rho^c}(\ln(i) \cup \{e'\})]$,

and further let $\tilde{f}_e = \begin{cases} \tilde{g}_e + 1_e & \text{if } \tilde{g}_e(e) = 0, \\ \tilde{g}_e(e)^{-1} \cdot \tilde{g}_e & \text{otherwise.} \end{cases}$

For those $CUT_{t,\rho}$ satisfying $e \in E_{t,\rho}$, update

$CUT_{t,\rho} = \{CUT_{t,\rho} \setminus \{e(t,\rho)\}\} \cup \{e\}$; and for others, $CUT_{t,\rho}$ remain unchanged.

end if

end for

Remark on Constructive Algorithm

Remark

- Our algorithm is a greedy one.

www.leaderstudio.net

Remark on Constructive Algorithm

Remark

- Our algorithm is a greedy one.
- If we choose $\beta_t = \delta_t$ for all $t \in \mathcal{T}$, then the proposed algorithm constructs a network MDS code.

www.leaderstudio.net

Remark on Constructive Algorithm

Remark

- Our algorithm is a greedy one.
- If we choose $\beta_t = \delta_t$ for all $t \in T$, then the proposed algorithm constructs a network MDS code.
- If we choose $\beta_t = 0$ for all $t \in T$, then this algorithm degenerates into an algorithm for constructing linear network codes and the required field size is $\sum_{t \in T} |R_t(\beta_t)| = \sum_{t \in T} |R_t(0)| = |T|$.

Comparing with other algorithms

- Our algorithm requires smaller base field than Matsumoto's and Yang *et al.*'s algorithms.

www.leaderstudio.net

Comparing with other algorithms

- Our algorithm requires smaller base field than Matsumoto's and Yang *et al.*'s algorithms.
- By randomly picking the encoding coefficients, our algorithm can be applied to both coherent and noncoherent cases.

www.leaderstudio.net

Comparing with other algorithms

- Our algorithm requires smaller base field than Matsumoto's and Yang *et al.*'s algorithms.
- By randomly picking the encoding coefficients, our algorithm can be applied to both coherent and noncoherent cases.
- Yang *et al.*'s algorithm designs the codebook and the local encoding kernels separately. On the contrary, Matsumoto's algorithm and our algorithm design them together.

www.leaderstudio.net

Comparing with other algorithms

- Our algorithm requires smaller base field than Matsumoto's and Yang *et al.*'s algorithms.
- By randomly picking the encoding coefficients, our algorithm can be applied to both coherent and noncoherent cases.
- Yang *et al.*'s algorithm designs the codebook and the local encoding kernels separately. On the contrary, Matsumoto's algorithm and our algorithm design them together.
- Our algorithm needs less storages at each sink node, compared with Matsumoto's algorithm.

Comparing with other algorithms

- The decoding of Matsumoto's algorithm requires exhaustive search by each sink node for all possible information from the source and all possible errors, and our algorithm can make use of the better and faster decoding algorithm — statistical decoding, and even the brute force decoding algorithm for our codes is better than the exhaustive decoding of Matsumoto's algorithm.

www.leaderstudio.net

Comparing with other algorithms

- The decoding of Matsumoto's algorithm requires exhaustive search by each sink node for all possible information from the source and all possible errors, and our algorithm can make use of the better and faster decoding algorithm — statistical decoding, and even the brute force decoding algorithm for our codes is better than the exhaustive decoding of Matsumoto's algorithm.
- If consider network error correction in packet networks, our codes may correct more errors beyond the error correction capability even minimum distance.

Example

- Topological order: $e_1 \prec e_2 \prec e_3$;
- $\omega = 1$, $\delta_t = C_t - \omega = 1$;
- $R_t(\delta_t) = R_t(1) =$
 $\{\rho_1 = \{e_1\}, \rho_2 = \{e_2\}, \rho_3 =$
 $\{e_3\}\}$;
- Base field: $\mathcal{F} = \mathbb{Z}_3$.

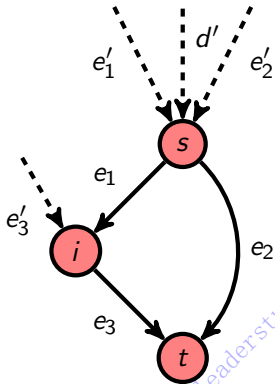


Figure: Network G_1 .

Example

$$\mathcal{P}_{t,\rho_1} = \{P_{t,\rho_1}^{(\delta_t)} = (e'_1, e_1, e_3),$$

$$P_{t,\rho_1}^{(\omega)} = (d', e_2)\};$$

$$\mathcal{P}_{t,\rho_2} = \{P_{t,\rho_2}^{(\delta_t)} = (e'_2, e_2),$$

$$P_{t,\rho_2}^{(\omega)} = (d', e_1, e_3)\};$$

$$\mathcal{P}_{t,\rho_3} = \{P_{t,\rho_3}^{(\delta_t)} = (e'_3, e_3),$$

$$P_{t,\rho_3}^{(\omega)} = (d', e_2)\}.$$

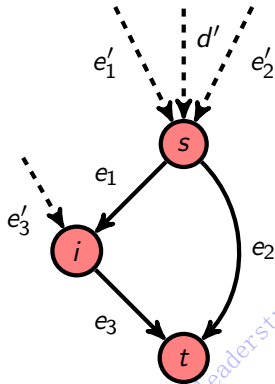


Figure: Network G_1 .

Example

Step 1: Initializing:

- $CUT_{t,\rho_1} = \{d', e'_1\}$,
 $CUT_{t,\rho_2} = \{d', e'_2\}$,
 $CUT_{t,\rho_3} = \{d', e'_3\}$;
- $\tilde{f}_{d'} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $\tilde{f}_{e'_1} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$,
- $\tilde{f}_{e'_2} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$, $\tilde{f}_{e'_3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$.

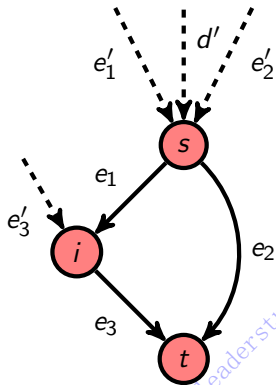


Figure: Network G_1 .

Example

Step 2: For the channel $e_1 \in Out(s)$, choose

$$\tilde{g}_{e_1} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \in \tilde{\mathcal{L}}(\{d', e'_1\}) \setminus$$
$$[\mathcal{L}^{\rho_1}(\{d'\}) + \mathcal{L}^{\rho_1^c}(\{d', e'_1\})] \cup [\mathcal{L}^{\rho_2}(\{e'_2\}) + \mathcal{L}^{\rho_2^c}(\{d', e'_1\})],$$

and let $\tilde{f}_{e_1} = \tilde{g}_{e_1}$, since $\tilde{g}_{e_1}(e_1) = 1$. Then update $CUT_{t, \rho_1} = \{d', e_1\}$,
 $CUT_{t, \rho_2} = \{e_1, e'_2\}$, and CUT_{t, ρ_3} remains unchanged.

www.leaderstudio.net

Example

Step 3: For the channel $e_2 \in Out(s)$, choose

$$\tilde{g}_{e_2} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \in \tilde{\mathcal{L}}(\{d', e'_2\}) \setminus [\mathcal{L}^{\rho_1}(\{e_1\}) + \mathcal{L}^{\rho_1^c}(\{d', e'_2\})] \\ \cup [\mathcal{L}^{\rho_2}(\{e_1\}) + \mathcal{L}^{\rho_2^c}(\{d', e'_2\})] \cup [\mathcal{L}^{\rho_3}(\{e'_3\}) + \mathcal{L}^{\rho_3^c}(\{d', e'_2\})],$$

and let $\tilde{f}_{e_2} = \tilde{g}_{e_2}$, since $\tilde{g}_{e_2}(e_2) = 1$. Then, update $CUT_{t, \rho_1} = \{e_2, e_1\}$,
 $CUT_{t, \rho_2} = \{e_1, e_2\}$, and $CUT_{t, \rho_3} = \{e_2, e'_3\}$.

www.leaderstudio.net

Example

Step 4: For the channel $e_3 \in Out(i)$, choose

$$\tilde{g}_{e_3} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} \in \tilde{\mathcal{L}}(\{e_1, e'_3\}) \setminus [\mathcal{L}^{\rho_1}(\{e_2\}) + \mathcal{L}^{\rho_1^c}(\{e_1, e'_3\})] \\ \cup [\mathcal{L}^{\rho_2}(\{e_2\}) + \mathcal{L}^{\rho_2^c}(\{e_1, e'_3\})] \cup [\mathcal{L}^{\rho_3}(\{e_2\}) + \mathcal{L}^{\rho_3^c}(\{e_1, e'_3\})],$$

and let $\tilde{f}_{e_3} = \tilde{g}_{e_3}$, since $\tilde{g}_{e_3}(e_3) = 1$. Then update
 $CUT_{t, \rho_1} = CUT_{t, \rho_2} = CUT_{t, \rho_3} = \{e_2, e_3\} \subseteq In(t)$.

www.leaderstudio.net

Example






The decoding matrix at t is $\tilde{F}_t = (\tilde{f}_{e_2} \ \tilde{f}_{e_3}) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$, which implies...

$$\Phi(t) \cap \Delta(t, \rho_i) = \{\underline{0}\}, \quad (i = 1, 2, 3) \Rightarrow d_{\min}^{(t)} = 2 = \delta_t + 1;$$





i.e., $\{\tilde{f}_{e_1}, \tilde{f}_{e_2}, \tilde{f}_{e_3}\}$ forms a global description of a network MDS code for the network G_1 .

www.leaderaudio.net

Reference





-  R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204-1216, Jul. 2000.
-  S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371-381, Jul. 2003.
-  R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782-795, Oct. 2003.
-  S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973-1982, Jun. 2005.
-  N. Cai and R. W. Yeung, "Network coding and error correction," in *Proc. IEEE Inform. Theory Workshop 2002*, Bangalore, India, Oct. 2002, pp. 119-122.

Reference

-  R. W. Yeung and N. Cai, "Network error correction, part I: Basic concepts and upper bounds," *Commun. Inf. Syst.*, vol. 6, pp. 19-36, 2006.
-  N. Cai and R. W. Yeung, "Network error correction, part II: Lower bounds," *Commun. Inf. Syst.*, vol. 6, pp. 37-54, 2006.
-  Z. Zhang, "Linear network error correction codes in packet networks," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, pp. 209-218, Jan. 2008.
-  S. Yang, R. W. Yeung, C. K. Ngai, "Refined coding bounds and code constructions for coherent network error correction," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1409-1424, Mar. 2011.

www.leader-tuo.net

Reference

-  R. Matsumoto, "Construction algorithm for network error-correcting codes attaining the singleton bound," *IEICE Trans. Fundamentals*, vol. E90-A, no. 9, pp. 1729-1735, Nov. 2007.
-  R. Koetter and F. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3579-3591, Aug. 2008.
-  D. Silva, F. Kschischang, and R. Kötter, "A rank-metric approach to error control in random network coding," *IEEE Trans. Inf. Theory*, vol. 54, no. 9, pp. 3951-3967, Sep. 2008.
-  Z. Zhang, "Theory and applications of network error correction coding," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 406-420, Mar. 2011.

Thanks for your attention!!!

Questions?

www.leaderstudio.net