Passive Network Tomography for Erroneous Networks: A Network Coding Approach

Hongyi Yao, *Member, IEEE,* Sidharth Jaggi, *Member, IEEE,* and Minghua Chen, *Member, IEEE,*

***Abstract*—Passive network tomography uses end-to-end observations of network communications to characterize the network, for instance, to estimate the network topology and to localize random or adversarial faults. Under the setting of linear network coding, this work provides a comprehensive study of passive network tomography in the presence of network (random or adversarial) faults. To be concrete, this work is developed along two directions: 1. Tomographic upper and lower bounds (*i.e.*, the most adverse conditions in each problem setting under which network tomography is possible, and corresponding schemes (computationally efficient, if possible) that achieve this performance) are presented for random linear network coding (RLNC). We consider RLNC designed with common randomness, i.e., the receiver knows the random code-books of all intermediate nodes. (To justify this, we show an upper bound for the problem of topology estimation in networks using RLNC without common randomness.) In this setting we present the first set of algorithms that characterize the network topology exactly. Our algorithm for topology estimation with random network errors has time complexity that is polynomial in network parameters. For the problem of network error localization given the topology information, we present the first computationally tractable algorithm to localize random errors, and prove it is computationally intractable to localize adversarial errors. 2. New network coding schemes are designed that improve the tomographic performance of RLNC while maintaining the desirable low-complexity, throughput-optimal, distributed linear network coding properties of RLNC. In particular, we design network codes based on Reed-Solomon codes so that a maximal number of adversarial errors can be localized in a computationally efficient manner even without the information of network topology. The tomography schemes proposed in the paper can be used to monitor networks with other faults such as packet losses and link delays, etc.**

***Index Terms*—Network coding, passive network tomography, network errors, adversaries.**

Hongyi Yao is with California Institute of Technology, Pasadena, CA 91125, USA, e-mail: yaohongyi03@gmail.com.

Sidharth Jaggi is with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin N.T., Hong Kong, e-mail:sidjaggi@gmail.com.

Minghua Chen is with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin N.T., Hong Kong, e-mail:minghua@ie.cuhk.edu.hk.

The major work of Hongyi Yao was done when he was a PhD student in the Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China, and was supported by the National Basic Research Program of China Grant 2007CB807900, 2007CB807901 and the National Natural Science Foundation of China Grant 61073174, 61033001, 61061130540. The work of Hongyi Yao was supported in part by NSF grant CNS-0905615. The work of Sidharth Jaggi was supported by the RGC GRF grants 412608 and 412809, the CUHK MoE-Microsoft Key Laboratory of Human-centric Computing and Interface Technologies, the Institute of Theoretical Computer Science and Communications, and Project No. AoE/E-02/08 from the University Grants Committee of the Hong Kong Special Administrative Region, China. The work of Minghua was partially supported by the China 973 Program 2012CB315904, the General Research Fund grants (Project No. 411008, 411009, 411010, and 411011) and an Area of Excellence Grant (Project No. AoE/E-02/08), all established under the University Grant Committee of the Hong Kong SAR, China, as well as two gift grants from Microsoft and Cisco.

I. Introduction

The goal of *passive network tomography* (or *passive network monitoring*) is to use end-to-end observations of network communication to infer the network topology, estimate link statistics such as loss rate and propagation delay, and locate network failures [3].

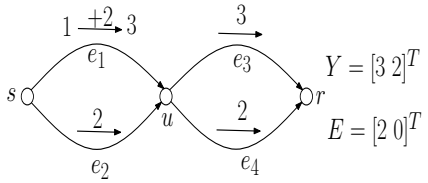
In networks using *linear network coding* each node outputs linear combinations of received packets; this has been shown to attain optimal multicast throughput [4]. In fact, even random linear network codes (where each node independently and randomly chooses the linear combinations to generate transmitted packets) suffice to attain the optimal multicast throughput [5], [8]. In addition to their desirable distributed nature, such schemes also have low design and implementation complexity [5], [8].

The main observation driving this work is that the linear transforms arising from random linear network coding have specific relationships with the network structure, and analyzing these relationships can significantly aid tomography. Prior works [9], [10] have also observed this relationship.

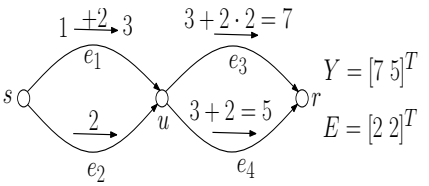
Toy example for error localization: Consider the tomography problem in Figure 1. Source s transmits probe symbols (say 1 and 2) to receiver r via intermediate node u. Suppose edge e_1 is erroneous and adds (say) 2 to every symbol transmitted over it. Receiver r knows the probe symbols, network, and communication schemes *a priori*. It also knows one of the links is erroneous (though it doesn't know in what manner), and wants to locate the erroneous link.

The case where the network communicates only via routing is shown in Figure 1(a). The probe symbols 1 and 2 are transmitted over edges e_1 and e_2 respectively to node u. Due to the error introduced over e_1, node u receives symbols 3 and 2 via edges e_1 and e_2 respectively, and forwards them to node r via edges e_3 and e_4 respectively. Node r receives two symbols from e_3 and e_4, denoted by a vector $Y = [3\ 2]^T$. Since r knows the probe symbols *a priori*, it can compute the error vector to be $E = Y - [1\ 2]^T = [2\ 0]^T$. Using E and its knowledge of the routing scheme, node r can infer that the error happened in the routing path $\{e_1, e_3\}$, but cannot figure out whether the error occurred on e_1 or e_3.

Figure 1(b) shows the case where node u applies linear network coding to transmit symbols. In particular, node u outputs $\mathbf{x_3} = \mathbf{x_1} + 2\mathbf{x_2}$ to link e_3 and $\mathbf{x_4} = \mathbf{x_1} + \mathbf{x_2}$ to e_4, where $\mathbf{x_1}$ and $\mathbf{x_2}$ are the symbols that node u receives from e_1 and e_2, and $\mathbf{x_3}$ and $\mathbf{x_4}$ are the symbols to be sent



(a) Routing Case



(b) Coding Case

Fig. 1. A tomographic example for locating an error at edge e_1. In Figure 1(a) observing error vector $E = [2\ 0]^T$ is not enough to distinguish the error locations e_1 and e_2. In Figure 1(b), since network coding is used by intermediate node u, the information of $E = [2\ 2]^T$ is enough to locate the erroneous edge e_1.

over e_3 and e_4. For a unit additive error $\mathbf{e} = 1$ at e_1, e_2, e_3 or e_4, the receiver r would observe error vectors $\mathbf{e}[1\ 1]^T$, $\mathbf{e}[2\ 1]^T$, $\mathbf{e}[1\ 0]^T$ or $\mathbf{e}[0\ 1]^T$, respectively. Thus, errors in different links result in observed error vectors in the corresponding vector spaces. Such linear algebraic characteristics of networks can be exploited to locate the erroneous link. Specifically, if error $\mathbf{e} = 2$ is injected into e_1, node r receives $Y = [7\ 5]^T$. Knowing in advance the probe symbols and node u's coding scheme, the receiver r computes the error vector as $E = Y - [1 + 2 \cdot 2, 1 + 2]^T = [2\ 2]^T$. Upon observing $E = [2\ 2]^T$ and comparing with the set of different error vector spaces corresponding to different error locations, r can determine that e_1 is the erroneous link and the error is $\mathbf{e} = 2$. □

While the toy example above might give the impression that the coding scheme needs to be carefully designed for the communication problem at hand, our results in this paper show that in fact random linear coding suffices for tomographic schemes that are distributed and have low computational and communication overhead. Further, if end-to-end network error-correcting codes (see for instance [11], [12]) are used for the network communications, in addition network tomography can *also* be implemented in a "passive" manner, *i.e.*, no dedicated probe messages are necessary. To be precise, the matrix received by the receiver in normal network communications contains sufficient information. With the help of end-to-end error correcting codes, the receiver can know the "error space" from such matrix. Thus throughout this work, the phrase "network tomography" stands for "passive network tomography" unless otherwise specified.

In this work we consider a network in which all nodes perform linear network coding. Besides receiving the messages, the receiver(s) wants to recover the network topology, and then detect and locate adversarial attacks, and random glitches (errors or erasures).

We perform a comprehensive study of passive network tomography in the presence of network errors, under the setting of network coding. In particular, we seek answers to the following questions.

1). In networks performing random linear network coding (RLNC), what are the appropriate tomographic upper and lower bounds? That is, what are the most adverse conditions in each problem setting under which network tomography is possible, and what schemes (computationally efficient, if possible) achieve this performance?

2). Are there any linear network coding schemes that improve upon the tomographic upper bounds for RLNC while maintaining their desirable low-complexity, throughput-optimal, distributed implementation properties?

A. Main contributions

We examine the relationship that linear transforms arising from random linear network coding have with the structure of the network. For this we find it useful to define the *impulse response vector* (IRV) $\mathbf{t}'(e)$ for every link e as the transform vector from link e to the receiver (see Section III-A for details). As shown in subsequent sections, each $\mathbf{t}'(e)$ can be treated as the fingerprint of corresponding link e. Any error on e exposes its fingerprint, allowing us to locate the error. Note that all the tomography schemes proposed in the paper for network errors can be used to monitor networks with other glitches such as network erasures (*i.e.*, packet losses) and link delays. We delay discussion on these related topics to the Appendix.

For network tomography under RLNC, our results are categorized into two classes:

Topology estimation. For networks suffering from random or adversarial errors, we provide the first algorithms that can estimate the network topology, given certain sufficient conditions are satisfied. In the case of random errors, our algorithms are computationally efficient. We also provide necessary conditions for such topology estimation to be possible. Our algorithms rely on the receivers (and the adversaries as well) have the access to a common randomness, *i.e.,* that the coding coefficients of each node are chosen from a random codebook known by the receiver. Without common randomness, we prove that in the presence of adversarial or random errors it is either theoretically impossible or computationally intractable to estimate topology accurately.

Error localization. We provide the first polynomial time algorithm for locating edges experiencing random errors. For networks suffering from adversarial errors we provide an upper bound of the number of locatable errors, and also a corresponding (exponential-time) algorithm that matches this bound. Moreover, we provide the first proof of computational intractability of the problem. Note that, as with error-localization schemes in the previous literature ([9], [14], [15], [16]), the schemes we provide for RLNC require the receiver to know the network topology and local linear coding coefficients – this can be from the topology estimation algorithms in this work, or as part of the network design *a priori*.

In the other direction, to circumvent the provable tomographic limitations of RLNC, we propose a specific class of random linear network coding that we call network Reed-Solomon coding (NRSC), which has the following desirable

features. NRSC is linear network coding that is implemented in a distributed manner (each network node only needs to know the node-IDs of its adjacent neighbors). With high probability over code design, NRSC achieves the the multicast capacity. Further, NRSC aids tomography in two aspects as follows.

- *Computational efficiency*. Under the adversarial error model, the receiver can locate adversarial errors in a computationally efficient manner. In particular, the number of locatable adversaries matches a corresponding tomographic upper bound. For the random error model, a lightweight topology estimation algorithm is provided under NRSC.
- *Robustness for dynamic networks*. For adversarial error localization the algorithms under NRSC do not require the prior knowledge of the network topology and thus are robust against dynamic network updating.

In Table I we compare our results and previous works on computational complexity.

B. Related work

Common randomness. Essentially all prior tomography results for RLNC assume some form of common randomness, *i.e.,*, the receiver is assumed to have some *a priori* knowledge about the random coding coefficients used by internal nodes. Some previous results [10], [9], [16] for locating errors under RLNC do not explicitly assume common randomness, but assume the receiver knows all the linear coding coefficients employed by each node in the network, which is related to our notion of common randomness.

We summarize related works on network inference under the following categories.

Passive tomography: The work in [10] provided the first explicit (exponential-time) algorithm for estimating the topology of an error-free network that performs RLNC. The work in [9] studied the problem of locating network errors for RLNC with the knowledge of network topology. In particular, error localization can be done in time $\mathcal{O}\left(\binom{|\mathcal{E}|}{z}\right)$, where $|\mathcal{E}|$ is the number of links in the network and z is the number of errors the network experiences.

Active tomography: The authors in [14], [15], [16] performed network tomography by using probe packets and exploiting the linear algebraic structure of network coding. The setting considered in these works concerns active tomography, whereas in this work we focus on passive tomography. For *active random error localization*, the authors in [14] and [16] studied error[1] localization in a network using *binary XOR* coding. Using *pre-designed* network coding and probe packets, they showed that the sources can use fewer probe packets than traditional tomography schemes based on routing. Again, $\mathcal{O}\left(\binom{|\mathcal{E}|}{z}\right)$ is the computational complexity of localization. For *active topology estimation*, using pre-designed binary XOR coding, the authors of [15] showed that the topology of a binary tree network can be recovered by using probe packets. The authors in [17] generalized the results to multi-source multi-receiver scenario.

Network inference with internal nodes' information: Another interesting set of works ([18], [19], [20]) inferred the network by the "packet information" of each internal node. In particular, using this internal "packet information", the work in [19] explored the subspace properties of packets received by internal nodes, the work in [18] proposed a scheme that infers the bottlenecks of P2P networks under network coding, and the works in [20], [21], [6] provided efficient schemes to locate the adversaries in the networks. Note that all these schemes require the receiver to know the information of the packets received by each internal node.

[1]In fact network erasures are considered in their works. Here we classify network erasures as a subclass of network errors.

C. Organization of the paper

The rest of this paper is organized as follows. We formulate the problem in Section II and present preliminaries in Section III. We then present our main technical results. Our results for network tomography consist of two parts. Part I considers RLNC, the schemes for topology estimation in the presence of network adversary and random errors are presented in Section IV, and the schemes for error localization is presented in Section V; Part II, consists of a particular type of RLNC, network Reed-Solomon coding (NRSC), in Sections VI, Section VII and Section VIII.

II. Problem Formulation and Preliminaries

A. Notational convention

Scalars are in lower-case (*e.g.* z). Matrices are in upper-case (*e.g.* X). Vectors are in lower-case bold-face (*e.g.* $\mathbf{e}$). Column spaces of a matrix are in upper-case bold-face (*e.g.* $\mathbf{E}$). Sets are in upper-case calligraphy (*e.g.* $\mathcal{Z}$).

B. Network setting

For ease of discussion, we consider a direct acyclic and delay-free network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Each node has a *unique identification number* known to itself. Such a label could correspond to the node's GPS coordinates, or its IP address, or a factory stamp. The capacity of each edge is normalized to be one symbol of $\mathbb{F}_q$ per unit time. We denote $e(u, v)$ as the edge from node u to v. In particular, u is said to be the *tail* node of e, and v is said to be the *head* node of e. For each node $v \in \mathcal{V}$, let $\mathbf{In}(v)$ be the set of all incoming edges (or nodes) of v and $\mathbf{Out}(v)$ be the set of all outgoing edges (or nodes) of v. The out-degree of node v is defined as $|\mathbf{Out}(v)|$ and in-degree of node v is defined as $|\mathbf{In}(v)|$.

Note that all the results in the paper can be generalized to the scenario where edges with non-unit capacity are allowed. A non-unit capacity edge is modeled as a set of parallel edges. This can be denoted by somewhat unwieldy notation, say $e(u, v, i)$, which stands for the i'th parallel edge from u to v.

We focus on the unicast scenario where a single source s communicates with a single receiver r over the network. In

TABLE I
COMPARISON OUR RESULTS AND PREVIOUS WORKS ON COMPUTATIONAL COMPLEXITY

Objective	Failure model	Tomography for RLNC [Previous works]	Tomography for RLNC [This work]	Tomography for NRSC [This work]
Topology Estimation	Adversarial Errors	-	Exponential	-
	Random Errors	-	Polynomial	Polynomial
Failure Localization	Adversarial Errors	Exponential [9]	Hardness Proof	Polynomial
	Random Errors	Exponential [9], [14]	Polynomial	Polynomial

principle, our results can be generalized to other communication scenarios where RLNC suffices. For instance, in networks with multiple receivers, we assume all incoming edges of the receivers are reconnected to a virtual receiver who performs network tomography.

Let C be the *min-cut* (or *max-flow*) from s to r. Without loss of generality, we assume that both the number of edges leaving the source s and the number of edges entering the receiver r equal C. We also assume that for every node in $\mathcal{V}$, there is at least one path from the node to the receiver r; Otherwise, the node is not involved in network communications, and hence is irrelevant to our study.

C. Dependency

Any set of z edges $e_1, e_2, ..., e_z$ is said to be *flow-independent* if there is a path from the head of each to the receiver r, and these z paths are edge-disjoint. The *flow-rank* of an edge-set $\mathcal{Z}$ equals the max-flow from the heads of edges in $\mathcal{Z}$ to the receiver r. A collection of edge-sets $\mathcal{Z}_1, \mathcal{Z}_2, ..., \mathcal{Z}_n$ is said to be *flow-independent* if *flow-rank*$(\cup_{i=1}^{n}\mathcal{Z}_i) = \sum_{i=1}^{n}$*flow-rank*$(\mathcal{Z}_i)$. The flow-rank of an internal node equals to the flow-rank of its outgoing edges. For the set $\mathcal{Z} \subseteq \mathcal{E}$ with flow-rank z, the *extended set* (or $Ext(\mathcal{Z})$) is the set that is of flow-rank z, includes $\mathcal{Z}$ and is of maximum size. Note that $Ext(\mathcal{Z})$ is well-defined and unique [22].

D. Network transmission via linear network coding

In this paper we consider the linear network coding scheme proposed in [23]. Let each packet have n symbols from $\mathbb{F}_q$, and each edge have the capacity of transmitting one packet, *i.e.*, a row vector in $\mathbb{F}_q^{1\times n}$.

Source encoder: The source s arranges the data into a $C\times n$ *message matrix* X over $\mathbb{F}_q$. Then on each outgoing edge of s, a linear combination over $\mathbb{F}_q$ of the rows of X is transmitted. Matrix X contains a pre-determined "short" *header* (say, the identity matrix in $\mathbb{F}_q^{C\times C}$) known in advance to both the source and the receiver, to indicate the linear transform from the source to the receiver.

Network encoders: Each internal node similarly takes linear combinations of the packets on incoming edges to generate the packets transmitted on outgoing edges. Let $\mathbf{x}(e)$ represent the packet traversing edge e. An internal node v generates its outgoing packet $\mathbf{x}(e')$ for edge $e' \in \mathbf{Out}(v)$ as

$$\mathbf{x}(e') = \sum_{e\in\mathbf{In}(v)} \beta(e, v, e')\mathbf{x}(e), \tag{1}$$

where $\beta(e, v, e')$ is the linear coding coefficient from the packet $\mathbf{x}(e)$ to the packet $\mathbf{x}(e')$ via v. As a default let $\beta(u, v, w) = \beta(e, v, e')$, where $e = (u, v)$ and $e' = (v, w)$.

Receiver decoder: The receiver r constructs the $C\times n$ matrix Y over $\mathbb{F}_q$ by treating the received packets as consecutive length-n row vectors of Y. The network's internal linear operations induce a linear transform between X and Y as

$$Y = TX, \tag{2}$$

where $T \in \mathbb{F}_q^{C\times C}$ is the overall transform matrix. The receiver r can extract T from the packet heads (recall that internal nodes mix heads in the same way as they mix messages). Once T is invertible the receiver can decode X as $X = T^{-1}Y$.

E. Network error models

Networks may experience disruption as a part of normal operation. Edge errors are considered in this work – node errors may be modeled as errors on edges outgoing from nodes with errors.

Let $\mathbf{x}(e) \in \mathbb{F}_q^{1\times n}$ be the input packet of e. For each edge $e \in \mathcal{E}$ a length-n row-vector $\mathbf{z}(e)$ is added into $\mathbf{x}(e)$. Thus the output packet of e is $\mathbf{y}(e) = \mathbf{x}(e)$+$\mathbf{z}(e)$. Edge e is said to suffer an error if and only if $\mathbf{z}(\mathbf{e})$ is a non-zero vector.

Both adversarial and random errors are considered.

- *Random errors*. Every edge e in $\mathcal{E}$ independently experiences random errors with a non-negative probability. A random error on e means that $\mathbf{z}(\mathbf{e})$ has *at least* one randomly chosen position, say i, such that the i'th symbol of $\mathbf{z}(\mathbf{e})$ is chosen from $\mathbb{F}_q$ uniformly at random. Note the difference of this model from the usual model of *dense random errors* on $\mathbb{F}_q$ [24], wherein $\mathbf{z}(\mathbf{e})$ is chosen from $\mathbb{F}_q^n$ at random. The model described in this work is more general in that it can handle such errors as a special case. However, it can *also* handle what we call "sparse" errors, wherein only a small fraction of symbols in $\mathbf{z}(\mathbf{e})$ are non-zero. Such a sparse error may be a more natural model of some transmission error scenarios [25], [26]. They may also be harder to detect. In our model we consider the worst-case sparsity of 1.
- *Adversarial errors*. The network is said to have z adversarial errors if and only if the adversary can arbitrarily choose a subset of edges $\mathcal{Z} \subseteq \mathcal{E}$ with $|\mathcal{Z}| = z$ and the corresponding erroneous packets $\{\mathbf{z}(e), e \in \mathcal{Z}\}$. Note that the adversary is assumed to have unlimited computational capability and to have the access to the information

of the source matrix X, network topology, all network coding coefficients and tomography algorithms used by the receiver.

F. Tomography Goals

The focus of this work is network end-to-end passive tomography in the presence of network errors. There are two tomographic goals. i) *Topology estimation*: The receiver r wishes to correctly identify the network topology upstream of it (*i.e.*, the graph $\mathcal{G}$). ii) *Error location*: The receiver r wishes to identify the locations where errors occur in the network.

In fact, all tomography schemes in the paper can be generalized in the following manner. Instead of the incoming edges $\mathbf{In}(r)$ of the receiver r, consider any cut $\mathcal{E}_C$ of edges that disconnects source s from receiver r. A network manager that has access to the packets output from $\mathcal{E}_C$ can use the tomography schemes in this paper to estimate the topology of the upstream network and locate the network errors.

G. Network error-correcting codes

Consider the scenario where a randomly or maliciously faulty set of edges $\mathcal{Z}$ of size z *injects* faulty packets into the network. As shown in [12], the network transform (2) then becomes into

$$Y = TX + E. \tag{3}$$

Note that the $C \times n$ *error matrix* E has rank at most z (see Section III-C for details). The goal for the receiver r in the presence of such errors is still to reconstruct the source's message X. Note that the loss-rate $2z/C$ is necessary and sufficient for correcting z adversarial errors [12], [11], while the loss-rate $(z+1)/C$ is necessary and sufficient [11] for correcting z random errors.

In this work we use the algorithms of [12] for correcting adversarial errors, and the algorithms of [11] for correcting random errors. All our tomography schemes use the performance guarantees provided by such end-to-end network error-correcting codes.

H. Computational hardness of NCPRLC

Several theorems we prove regarding the computational intractability of some tomographic problems utilize the hardness results of the following well-studied problem.

The Nearest Codeword Problem for Random Linear Codes (NCPRLC) is defined as follows:

- *NCPRLC:* $(H, z, \mathbf{e})$: Given a *parity check matrix* H which is chosen uniformly at random over $\mathbb{F}_q^{l_1 \times l_2}$ with $l_2 > l_1$, a constant z, and a vector $\mathbf{e} \in \mathbf{H}$ which is linear combined from at most z columns of H. The algorithm is required to output a *z-sparse solution* $\mathbf{b}$ (*i.e.,* $\mathbf{b}$ has at most z nonzero components) such that $\mathbf{e} = H\mathbf{b}$.

Note that NCPRLC is a well-known computationally hard problem [27], [28].

I. Decoding of Reed-Solomon codes

This section introduces some properties of the well-studied Reed-Solomon codes (RSCs) [29], used in particular for worst-case error-correction for point-to-point channels. A Reed Solomon code (RSC) is a linear error-correcting code over a finite field $\mathbb{F}_q$ defined by its parity check matrix $H \in \mathbb{F}_q^{l_1 \times l_2}$ with $l_2 > l_1$. Here $l_1 + 1$ is the *minimum Hamming distance* of the code, *i.e.*, minimum number of nonzero components among the codewords belonging to the code. In particular, H is formed as

$$H = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{l_2}], \tag{4}$$

where $\mathbf{h}_i = [h_i, (h_i)^2, ..., (h_i)^{l_1}]^T \in \mathbb{F}_q^{\ell_1}$ and $h_i \neq 0$ for each $i \in [1, l_2]$ and $h_i \neq h_j$ for $i \neq j$.

Given $\mathbf{e}$ which is a linear combination of any $z \leq (l_1+1)/2$ columns of H, the decoding algorithm of RS-CODE, denoted as $\mathbf{RS\text{-}DECODE}(H, \mathbf{e})$, outputs a *$z$-sparse solution* of $H\mathbf{b} = \mathbf{e}$ with $O(l_2 l_1)$ operations over $\mathbb{F}_q$ (see [30]). That is, $\mathbf{b} \in \mathbb{F}_q^{l_2}$ has at most z non-zero components and $\mathbf{e} = H\mathbf{b}$. Further more, for any $\mathbf{b}' \neq \mathbf{b}$, either $\mathbf{e} \neq H\mathbf{b}'$ or $\mathbf{b}'$ has more than z non-zero components, *i.e.*, $\mathbf{b}$ is the unique z-sparse solution of $H\mathbf{b} = \mathbf{e}$.

III. Impulse Response Vector (IRV)

In this section, we explain the relationship between the linear transforms induced by the linear network coding and the network structures, by introducing the concept of *impulse response vector* (IRV). The relationship forms the mathematical basis for our proposed tomography schemes.

A. Definition of Impulse Response Vector (IRV)

Corresponding to each edge $e \in \mathcal{E}$ we define the length-C impulse response vector (IRV) $\mathbf{t}'(e) \in \mathbb{F}_q^C$ as the linear transform from e to the receiver. In particular, let the source s transmit the *all-zero packet* $\mathbf{0} \in \mathbb{F}_q^n$ on all outgoing edges, let edge e inject a packet $\mathbf{z}(e) \in \mathbb{F}_q^n$, and let each internal node perform linear network coding operations. Then the matrix Y received by the receiver r is $Y = \mathbf{t}'(e)\mathbf{z}(e) \in \mathbb{F}_q^{C \times n}$. So $\mathbf{t}'(e)$ can be thought of as a "unit impulse response" from e to r.

Illustrating examples for edge IRVs are in Figures 2 and 3. In Figure 2, coding coefficients for a unit-length packet are shown. In Figure 3(a), only e_4 has an injected symbol 1 and what r receives is $Y = [1, 0]^T$, thus the IRV of e_4 is $\mathbf{t}'(e_4) = [1, 0]^T$. For the same reason, the IRVs of e_5, e_3, e_2 and e_1 are computed similarly in Figure 3(b) , Figure 3(c), Figure 3(d) and Figure 3(e) respectively.

For a set of edges $\mathcal{Z} \subseteq \mathcal{E}$ with $|\mathcal{Z}| = z$, the columns of the $C \times z$ *impulse response matrix* $T'(\mathcal{Z})$ comprise of the set of vectors $\{\mathbf{t}'(e) : e \in \mathcal{Z}\}$.

All IRVs can be inductively computed. First, the IRV for each edge incoming to the receiver is set as a distinct unit vector, *i.e.*, a distinct column of the $C \times C$ identity matrix. Then for each edge e incoming to node v with outgoing edges $\{e_1, e_2, ..., e_d\}$, we have

$$\mathbf{t}'(e) = \sum_{j=1,2,...,d} \beta(e, v, e_j)\mathbf{t}'(e_j).$$

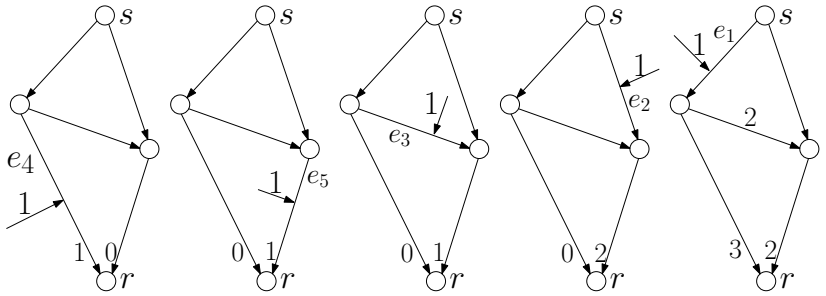


Fig. 3. The IRVs of the edges shown in Figure 2: $\mathbf{t}'(e_4) = [1, 0]^T$, $\mathbf{t}'(e_5) = [0, 1]^T$, $\mathbf{t}'(e_3) = \mathbf{t}'(e_5) = [0, 1]^T$, $\mathbf{t}'(e_2) = 2\mathbf{t}'(e_5) = [0, 2]^T$ and $\mathbf{t}'(e_1) = 3\mathbf{t}'(e_4) + 2\mathbf{t}'(e_3) = [3, 2]^T$. Edges e_2 and e_3 are not flow-independent, so the IRV $\mathbf{t}'(e_2)$ equals the $\mathbf{t}'(e_3)$ (up to a scalar multiple). Conversely, e_1 and e_5 are flow-independent, so $\mathbf{t}'(e_1)$ is linearly independent from $\mathbf{t}'(e_5)$.

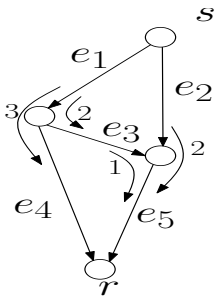


Fig. 2. An example network and its local coding coefficients.

B. IRVs under random linear network coding (RLNC)

The linear network coding defined in Section II-D is a random linear network coding (RLNC) if and only if [7] the following conditions are true.

Source encoder: The source s takes C independently and uniformly random linear combinations of the rows of X to generate respectively the packets transmitted on each edge outgoing from s (recall that exactly C edges leave the source s).

Network encoders: Each internal node, say v, independently and uniformly chooses its local coding coefficients $\{\beta(e, v, e'), e \in \mathbf{In}(v), e' \in \mathbf{Out}(v)\}$ at random.

Receiver decoder: As shown in Equation (2), the receiver r receives Y as $Y = TX$, where T is the overall transform matrix. It is proved that with a probability at least $1-|\mathcal{E}|/q$, the matrix T is invertible under RLNC [7]. The receiver extracts T from the header of Y and decodes X as $X = T^{-1}Y$.

For RLNC, the linear transforms defined in Section III-A provide algebraic interpretations for the graphs. To be concrete, Lemma 1 below states that the linear independence of the IRVs has a close relationship with the flow-independence of the edges. This relationship is used in tomography schemes shown in later sections.

Lemma 1: 1) The rank of the impulse response matrix $T'(\mathcal{Z})$ of an edge set $\mathcal{Z}$ with flow-rank z is at most z.
2) The IRVs of flow-independent edges are linearly independent with probability at least $1 - |\mathcal{E}|/q$.

Proof:

1) When the flow-rank of $\mathcal{Z}$ is z, the max-flow from $\mathcal{Z}$ to r is at most z. If the rank of $T'(\mathcal{Z})$ is larger than z, say $z+1$, $\mathcal{Z}$ can transmit information to r at rate $z+1$, which is a contradiction.
2) For an flow-independent edge set $\mathcal{Z}$ with cardinality z, assume a virtual source node s' has z virtual edges connected to the tails of $\mathcal{Z}$, and all outgoing edges (except for $\mathcal{Z}$) of the tails of $\mathcal{Z}$ are deleted. The max-flow from s' to r is z and $\mathcal{Z}$ is a cut. Then $T'(\mathcal{Z})$ has rank z if and only if s' can transmit information to r at rate z. By a direct corollary of Theorem 1 in [5], this happens with probability at least $1 - |\mathcal{E}|/q$.

□

Thus for a large enough field-size q, properties of the network edges map to similar properties of the IRVs. For instance, assume $\mathcal{Z}_1, \ldots, \mathcal{Z}_d$ are edge sets. Using the Union Bound over the edge sets, with probability at least $1 - (d + 1)|\mathcal{E}|/q$, we have *flow-rank*$(\mathcal{Z}_i) = rank(T'(\mathcal{Z}_i))$ for any $i \in \{1, \ldots, d\}$, and *flow-rank*$(\cup_{i=1}^{d} \mathcal{Z}_i) = rank(T'(\cup_{i=1}^{d} \mathcal{Z}_i))$. Thus, *flow-rank*$(\cup_{i=1}^{d} \mathcal{Z}_i) = \sum_{i=1}^{d}$ *flow-rank*$(\mathcal{Z}_i)$ if and only if $rank(T'(\cup_{i=1}^{d} \mathcal{Z}_i)) = \sum_{i=1}^{d} rank(T'(\mathcal{Z}_i))$. Thus by studying the ranks of $\{T'(\mathcal{Z}_1), \ldots, T'(\mathcal{Z}_d)\}$, we can infer relationships between the topological structures of $\{\mathcal{Z}_1, \ldots, \mathcal{Z}_d\}$.

The example in Figure 3 also shows the relationship between flow-independence and linear independence.

C. IRVs for network errors

Assume a faulty set of edges $\mathcal{Z}$ of size z *injects* faulty packets into the network, *i.e.*, $\mathcal{Z} = \{e : e \in \mathcal{E}, \mathbf{z}(e) \neq 0\}$ and $|\mathcal{Z}| = z$. From the definition of IRV, we have

$$Y = TX + T'(\mathcal{Z})Z, \tag{5}$$

where Z is a $z \times n$ matrix whose rows comprise of additive error packets $\{\mathbf{z}(e) : e \in \mathcal{Z}\}$. Thus the error matrix E defined in Equation (3) (of Section II-G) equals $T'(\mathcal{Z})Z$ and has rank at most z.

IV. Topology estimation for RLNC

In the first part of our technical results, we construct schemes for topology estimation in the presence of network adversary and random errors in Section IV, and schemes for error localization in Section V.

A. Common randomness

Common randomness means that all candidate local coding coefficients $\{\beta(u,v,w), u,w \in \mathcal{V}\}$ of node $v \in \mathcal{V}$ are chosen from its local random code-book $\mathcal{R}_v$, and the set of all local random code-books $\mathcal{R} = \{\mathcal{R}_v, v \in \mathcal{V}\}$ is known *a priori* to the receiver r. Note that $\mathcal{R}$ can be public to all parties including the adversaries.

Common randomness is both necessary and sufficient for network topology estimation under RLNC. On one hand sufficiency follows from the results of [10] and those of this section. On the other hand we show that in the presence of adversarial (or random errors), determining the topology without assuming common randomness is theoretically impossible (or computationally intractable) in Theorem 2 and Theorem 3.

Each local random code-book in $\mathcal{R}$ comprises of a list of elements from $\mathbb{F}_q$, with each element chosen independently and uniformly at random. These random code-books can be a part of network design, or computed by using pseudorandom function with node ID as the input.

Depending on the types of failures in the network, we define two types of common randomness. Recall that $\beta(u,v,w)$ is the local coding coefficient from edge $e(u,v)$ via v to the edge $e'(v,w)$ (see Section II-D for details).

Weak type common randomness for random errors. For node $v \in \mathcal{V}$ each distinct element (u,w) in $\mathcal{V} \otimes \mathcal{V}$ indexes a distinct element in $\mathcal{R}_v$. The local coding coefficient $\beta(u,v,w)$ is chosen as the element $\mathcal{R}_v(u,w)$. For instance consider the subnetwork shown in Figure 4. Under weak type common randomness[2], Figure 5 shows how node v_1 chooses the coding coefficient $\beta(v_2,v_1,v_4)$.

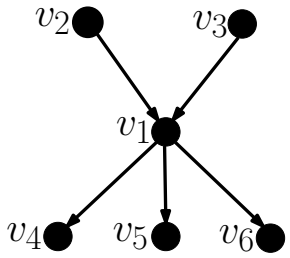


Fig. 4. The adjacent neighbors of node v_1.

Strong type common randomness for adversarial errors. For node $v \in \mathcal{V}$ each distinct element (u,w,w') in $\mathcal{V} \otimes \mathcal{V} \otimes \mathcal{V}$ indexes a distinct element in $\mathcal{R}_v$. For an instance network, recall that $\mathbf{Out}(v)$ is set of the outgoing edges of v. The coding coefficient $\beta(u,v,w)$ is chosen as

$$\beta(u,v,w) = \sum_{e(v,w') \in \mathbf{Out}(v)} \mathcal{R}_v(u,w,w'). \qquad (6)$$

For instance consider the subnetwork shown in Figure 4. Under strong type common randomness, Figure 6 shows how node v_1 chooses the coding coefficient $\beta(v_2,v_1,v_4)$.

[2]We note that for network with parallel edges the random code-book $\mathcal{R}_v$ can be described by somewhat unwieldy notation. For instance, under weak common randomness the element $\mathcal{R}_v(u,w,i,j)$ is for the coding coefficient from edge (u,v,i) (*i.e.*, the ith parallel edge between u and v) to (v,w,j) via v.

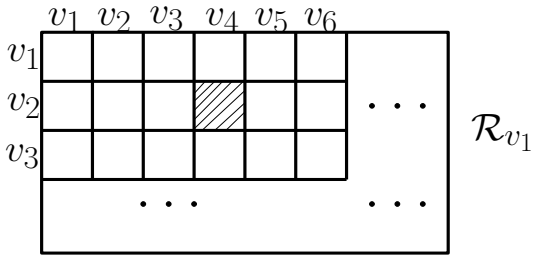


Fig. 5. Under weak type common randomness, node v_1 in Figure 4 chooses $\beta(v_2,v_1,v_4)$ as the element shown in the dark region.

For adversarial errors it is required that the existence of an edge $e(v,w)$ would affect the particular choice of the coding coefficients $\{\beta(u,v,w') : w' \neq w\}$. Otherwise, if the adversary corrupts $e(v,w)$ and only sends all-zero packet on $e(v,w)$, it is impossible for the receiver to notice the existence of $e(v,w)$. Thus a different type of common randomness is used for networks suffering adversarial errors. For such strong type common randomness, since 1) all symbols in $\mathcal{R}_v$ are independently and uniformly chosen over finite field $\mathbb{F}_q$ and 2) for each coding coefficient $\beta(u,v,w)$ the summation in equation (6) involves distinct elements in $\mathcal{R}_v$, the coding coefficient $\beta(u,v,w)$ chosen by equation (6) is also independently and uniformly distributed over $\mathbb{F}_q$.

We first prove the necessity of using common randomness for topology estimation in networks with adversarial errors. Since the network adversaries can hide themselves and only inject zero errors, it suffices to prove common randomness is necessary for topology estimation in error-free networks.

Theorem 2: If internal nodes choose local coding coefficients independently and randomly *without* assuming common randomness, there exist two networks which cannot be distinguished by the receiver in the absence of network errors.

Proof: Since the overall transform matrix (see Equation (2) for details) is the only information the receiver can retrieve from the receiving packets, it suffices to prove the overall transform matrixes of $Exp1$ and $Exp2$ in Figure 7 are statistically indistinguishable.

For both $Exp1$ and $Exp2$, the transform matrixes $T(1)$ and $T(2)$ are in fact single elements over $\mathbb{F}_q$. For $Exp1$, let $a \in \mathbb{F}_q$ be the transform coding coefficient from s to u_1. Similarly, $[b_1,b_2]^T \in \mathbb{F}_q^{2\times 1}$, $[c_1,c_2] \in \mathbb{F}_q^{1\times 2}$, and $d \in \mathbb{F}_q$ are the transform coding coefficients from u_1, u_2, u_3 to the adjacent downstream nodes respectively. Thus, the overall transform matrix $T(1)$ from s to r in $Exp1$ is $T(1) = d[c_1,c_2][b_1,b_2]^T a = ab_1c_1d + ab_2c_2d$.

Similarly, let $a' \in \mathbb{F}_q$ be the transform coding coefficient from s to v_1. Similarly, $b' \in \mathbb{F}_q$, $[c_1',c_2']^T \in \mathbb{F}_q^{2\times 1}$, and $[d_1',d_2'] \in \mathbb{F}_q^{1\times 2}$ are the transform coding coefficients from v_1, v_2, v_3 to the adjacent downstream nodes respectively. Thus, the overall transform matrix $T(2)$ from s to r in $Exp2$ is $T(2) = [d_1',d_2'][c_1',c_2']^T b'a' = a'b'c_1'd_1' + a'b'c_2'd_2'$.

Since each element of $\{a, b_1, b_2, c_1, c_2, d, a', b', c_1', c_2', d_1', d_2'\}$ is independently and uniformly chosen at random, $T(1)$ is statistically indistinguishable from $T(2)$. □

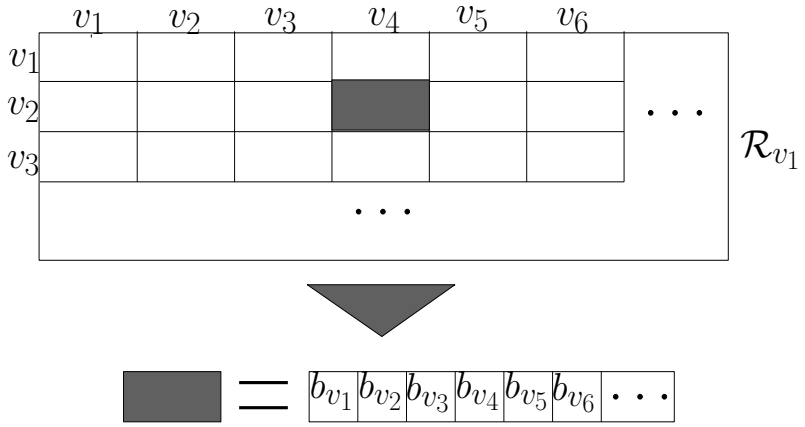


Fig. 6. Under strong type common randomness, node v_1 in Figure 4 chooses $\beta(v_2, v_1, v_4)$ as $b_{v_4} + b_{v_5} + b_{v_6}$.

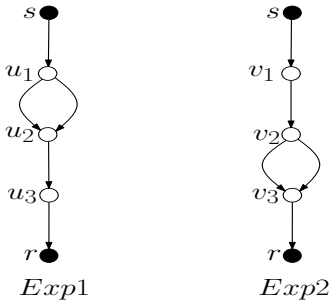


Fig. 7. Two networks that are impossible to distinguish by the receiver.

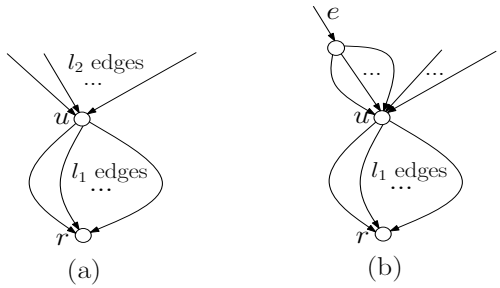


Fig. 8. A network reduced from the NCPRLC instance $(H, z, \mathbf{e})$.

For the random error model (see Section II-E for details), Theorem 2 does not suffice to show the necessity of common randomness. The reason is that in a zero error network the only network information observed by the receiver is the transform matrix T, while in networks suffering random errors, a random error on the edge may expose its IRV information which aids topology estimation. In the following, it is proved that without assuming common randomness topology estimation is at least as computationally intractable as NCPRLC (see the definition in Section II-H for details).

For notational convenience, we define $\mathcal{I}_{IRV}$ to be a set of vectors, *i.e.*, a subset of $\mathbb{F}_q^C$. Set $\mathcal{I}_{IRV}$ is a collection of IRVs of all edges in the network. Note that $\mathcal{I}_{IRV}$ is merely a set of vectors, and as such, individual element has no correspondence with any edge in the network. For instance, for the network in Fig 3, $\mathcal{I}_{IRV}$ is defined as $\{[0,1]^T, [0,2]^T, [1,0]^T, [3,2]^T\}$.

For the random error model, as in (5), the receiver gets $Y = TX + E$, where $E = T'(\mathcal{Z})Z$. Thus E and T are all the information observed by the receiver r. When the edges suffer random errors independently, since the errors in Z are uniformly random, the error matrix $E = T'(\mathcal{Z})Z$ cannot provide more information than $T'(\mathcal{Z})$, whose columns are in $\mathcal{I}_{IRV}$. Thus it suffices to prove:

Theorem 3: When the internal nodes choose local coding coefficients independently and randomly *without* assuming common randomness, if the receiver r can correctly output the topology in polynomial time (in network parameters) with the knowledge of T and $\mathcal{I}_{IRV}$, NCPRLC can be solved in polynomial time (in problem parameters).

Proof: Given a NCPRLC instance $(H, z, \mathbf{e})$, as shown in Figure 8(a), we construct a network with l_1 edges to r and l_2 edges to node u.

Since H is a matrix chosen uniformly at random over $\mathbb{F}_q^{\ell_1 \times \ell_2}$, it corresponds to a RLNC, where each column of H corresponds to an IRV of an edge in $\mathbf{In}(u)$.

Since $\mathbf{e}$ is a linear combination of z columns of H, we can assume that there is an edge e that is connected to z edges in $\mathbf{In}(u)$ (see Figure 8(b)), and that the IRV of e is $\mathbf{e}$. If the receiver r can recover the topology, r is able to tell which z edges in $\mathbf{In}(u)$ are connected to e. Thus r can find a linear combination of z columns of H resulting in $\mathbf{e}$ and thus solve $(H, z, \mathbf{e})$. □

B. Topology estimation for networks with adversarial errors

In this section, we use an error-correcting code approach [12] to estimate the topology of a network with adversarial errors. At a high level, the idea is that in strongly connected networks (which is defined in Assumption 2 below), each pair of networks generates transform matrices that look "very different". Hence no matter what the adversary does, he is unable to make the transform matrix for one network

resemble that of any other. The estimation algorithm and proof techniques are similar in flavor to those from algebraic coding theory.

As is common in the network error-correcting literature (for instance, [11], [12]), we assume that the adversary is bounded, and therefore corrupts no more than z edges in the network.

Assumptions and justifications:

1) At most z edges in $\mathcal{Z}$ suffer errors, *i.e.*, $\{e : e \in \mathcal{E}, \mathbf{z}(e) \neq 0\} = \mathcal{Z}$ and $|\mathcal{Z}| \leq z$. When $2z+1 \leq C$, network-error-correcting codes (see Section II-G for details) are used so that the source message X is provably decodable.
2) *Strong connectivity*. A set of networks satisfies "strong connectivity" if the following is true. Each internal node has both in-degree and out-degree at least $2z+1$. Note that in an acyclic graph it implies that the source has at least $2z+1$ edge disjoint paths to each internal node, which has $2z+1$ edge disjoint paths to the receiver. We motivate this strong connectivity requirement by showing in Theorem 6 lower bounds on the connectivity required for *any* topology estimation scheme to work in the presence of an adversary.[3]
3) *Knowledge of local topology.* We assume that each node knows the ID numbers of the nodes exactly one hop away from it, either upstream or downstream of it.
4) *Strong type common randomness is assumed.* This assumption is justified by Theorem 2.

After receiving the overall transform matrix T_e which is polluted by the network adversarial errors, the receiver r uses the following algorithm to estimate of topology of the network. The algorithm below essentially finds a matrix that is "closest" to the observed matrix in a certain metric (details are below), but is "feasible", *i.e.*, corresponds to a transfer matrix for an error-free network. It then estimates the topology as the network corresponding to this closest matrix.

Algorithm 1 TOPO-ADV-RLNC

Input: Matrix T_e and codebook $\mathcal{R} = \{\mathcal{R}_v, v \in \mathcal{V}\}$

1: **for** Each candidate graph $\mathcal{G}$ satisfying the strong connectivity requirement **do**
2: Using $\mathcal{R}$, compute the overall transform matrix $T(\mathcal{G})$ for $\mathcal{G}$
3: **if** $\text{rank}(T_e - T(\mathcal{G})) \leq z$ **then**
4: **return** $\mathcal{G}$
5: **end if**
6: **end for**

Before proving the correctness of **TOPO-ADV-RLNC** we show the key lemma for the *rank distance* of different graphes. The *rank-distance* between any two matrices $A, B \in \mathbb{F}_q^{C\times C}$ is defined as $r_m(A,B) = \text{rank}(A-B)$. We note that rank-distance indeed satisfies the properties of a distance function; in particular it satisfies the triangle inequality [12].

Lemma 4: Let the transform matrices of two different networks $\mathcal{G}$ and $\mathcal{G}'$ be $T(\mathcal{G})$ and $T(\mathcal{G}')$ respectively. Then with probability at least $1-|\mathcal{V}|^4/q$, $r_m(T(\mathcal{G}), T(\mathcal{G}')) \geq 2z+1$. With probability at least $1-|\mathcal{V}|^4 2^{|\mathcal{V}|^2}/q$, it holds for any pair of networks [4].

Proof: Without loss of generality, we assume that there exists a node $u \neq r$ such that v has an outgoing edge e_v in $\mathcal{G}$ but not in $\mathcal{G}'$.

We first show that there exists a $(2z+1)\times(2z+1)$ sub-matrix in $T(\mathcal{G}) - T(\mathcal{G}')$, such that its determinant is not zero on an realization of the elements of the code-books in $\mathcal{R}$. Using Assumption 2), in $\mathcal{G}$ there exist $2z+1$ edge disjoint paths from s to r via v. The elements in $\mathcal{R}$ can be evaluated such that i) only the routing transmissions along these paths are allowed for any graph over node set $\mathcal{V}$; ii) in $\mathcal{G}$, the source s can transmit $2z+1$ packets using routing via v to r; iii) the elements in $\mathcal{R}_v$ satisfy $\mathcal{R}_v(u,w,w') \neq 0$ only if $(v,w') = e_v$.

Thus for graph $\mathcal{G}$, under such a realization of $\mathcal{R}$ the transform matrix $T(\mathcal{G})$ has a sub-matrix as a $(2z+1)\times(2z+1)$ identity matrix.

Since $e_v \notin \mathcal{G}'$, due to condition iii) above and the definition of strong type common randomness, in $\mathcal{G}'$ all local coding coefficients used by v are zero. Due to condition i) above, only routing transmissions via node v are allowed. Thus the transform matrix $T(\mathcal{G}')$ for graph $\mathcal{G}'$ is a zero matrix.

Thus under such a realization of $\mathcal{R}$, $T(\mathcal{G}) - T(\mathcal{G}')$ has a $(2z+1)\times(2z+1)$ sub-matrix with determinant 1.

Each element of such a sub-matrix is a polynomial of local coding coefficients, and therefore a polynomial of random variables belonging to $\mathcal{R}$. The degree of such polynomial in no more than $|\mathcal{E}|$. Thus, the determinant of the sub-matrix is a polynomial of random variables in $\mathcal{R}$, with degree at most $|\mathcal{E}|\times(2z+1) \leq |\mathcal{E}|^2 \leq |\mathcal{V}|^4$. Using Schwarth-Zippel Lemma [32], with probability at least $1-\frac{|\mathcal{V}|^4}{q}$ the determinant of the sub-matrix is nonzero, i.e., $r_m(T(\mathcal{G})-T(\mathcal{G}')) \geq 2z+1$.

Since there are at most $2^{|\mathcal{V}|^2/2}$ acyclic graphs and $2^{|\mathcal{V}|^2}$ pairs of graphs to be compared, following a Union Bound [33] argument, with probability at least $1-|\mathcal{V}|^4 2^{|\mathcal{V}|^2}/q$, it holds for any pair of networks. □

As in (5), after transmission, the erroneous transfer matrix T_e received by r is actually

$$T_e = T + T'(\mathcal{Z})Z_h, \tag{7}$$

where Z_h represents the errors injected for the packet headers, *i.e.*, the first C columns of Z. This combined with Lemma 4 enables us to prove the correctness of **TOPO-ADV-RLNC**.

Theorem 5: With probability at least $1-|\mathcal{V}|^4 2^{|\mathcal{V}|^2}/q$, the network $\mathcal{G}$ outputted by **TOPO-ADV-RLNC** is the correct network.

Proof: We assume Lemma 4 is true for any pair of graphs, which happens with probability at least $1-|\mathcal{V}|^4 2^{|\mathcal{V}|^2}/q$ as stated above.

[3]Note that for the single source (or single receiver) network, such connectivity requires parallel edges at the source (or the receiver). Otherwise if parallel edges are not allowed, we assume the neighbors of the source (or the receiver) are the end-nodes, *i.e.*, they are not in the domain of tomography. Similar argument holds in later sections.

[4]For counting the total number of networks we do not count the the networks with parallel edges for clarity of exposition. When parallel edges are taken into account, the number of bits required to represent the field-size q should be $\Theta(|\mathcal{V}|^2\log(|\mathcal{E}|))$, so as to make the probability of failure of the tomography algorithm negligible..

By (7), the rank distance $r_m(T_e, T(\mathcal{G}))$ equals $rank(T'(\mathcal{Z})Z_h) \leq rank(T'(\mathcal{Z})) \leq z$. For any transfer matrix $T(\mathcal{G}')$ corresponding to a different network $\mathcal{G}'$, by the triangle inequality of the rank distance, $r_m(T(\mathcal{G}'), T_e) \geq r_m(T(\mathcal{G}'), T(\mathcal{G})) - r_m(T(\mathcal{G}), T_e) \geq z + 1$. This completes the proof. □

Finally, we show that the strong connectivity requirements (see Assumption 2) for details) we require for Theorem 5 are "almost" tight. We remark that there is a mismatch between the sufficient connectivity requirement in Assumption 2 (each internal node has in-degree at least $2z+1$), and the necessary connectivity requirement of Theorem 6 (each internal node has in-degree at least $z + 1$). Whether the gap between such mismatch can be closed is still open. We note that Theorem 6 holds for any network transmission scheme and network tomography scheme.

Theorem 6: For any network $\mathcal{G}$ that has a node with in-degree less than than $z + 1$, or a node with out-degree less than $2z + 1$, there exists an adversarial action that makes any tomographic scheme fail to estimate the network topology.

Proof: Assume node v has $2z$ outgoing edges, and the adversary controls a set $\mathcal{Z}$ of size z of them. Let $\mathcal{Z}'$ be the other z outgoing edges of v that are not corrupted by the adversary. If an adversary transmits messages on $\mathcal{Z}$ claiming that a node the node at the tails is u (different from v), the receiver cannot distinguish this from the case in which the adversary corrupts the edges in $\mathcal{Z}'$ and claims that the node at the tail is v (whereas the actual node in this alternate scenario is u).

On the other hand, if v has only z incoming edges, the adversary can cut these off (*i.e.* simulate erasures on these edges). Since the node can only transmit the message from its incoming edges, this implies that all messages outgoing from v are also, essentially, erased. Hence the presence of v cannot be detected by r. □

In fact, the proof of Lemma 4 only requires that $\mathcal{G}$ and $\mathcal{G}'$ differ at a node with high connectivity. If we know the possible topology set *a priori*, we can relax the connectivity requirement. The following corollary formalizes the observation.

Corollary 7: For a set of candidate networks $\{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_d\}$, if any two of them differ at a node which has max-flow at least $2z + 1$ from the source and max-flow at least $2z + 1$ to the receiver, with a probability at least $1 - d^2|\mathcal{V}|^4/q$ the receiver can find the correct topology by the receiving transform matrix.

Proof: Following the proof of Lemma 4, we conclude that for any two distinct graphs in $\{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_d\}$, with probability at least $1-|\mathcal{V}|^4/q$, their transform matrixes have rank distance at least $2z + 1$. Using the union bound, with probability at least $1 - d^2|\mathcal{V}|^4/q$, the transform matrixes of every pair of graphs in $\{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_d\}$ have rank distance at least $2z + 1$. Following the proof of Theorem 5, with probability at least $1 - d^2|\mathcal{V}|^4/q$, the network $\mathcal{G}$ output by **TOPO-ADV-RLNC** is the correct network. ■

We note that this corollary reduces the connectivity requirement for practical tomography scenarios. In practice it is easier to know the network topology "near" you, of network resources that you control, than to know the internal network topology of "public networks" controlled by independent entities (such as ISPs connecting different local networks). Assuming the topology of the local network is known, candidate networks $\{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_d\}$ are those differing at such public networks. Further, nodes in public networks are more likely to have high degrees [13], which makes Corollary 7 easier to satisfy.

C. Topology estimation for networks with random errors

Under RLNC, we provide a polynomial-time scheme to recover the topology of the networks that suffer from random network errors (the definition of random errors can be found in Section II-E). The receiver r proceeds in two stages. In the first stage (**Algorithm 2 FIND-IRV**), r recovers the IRV information during several rounds of network communications suffering random errors. In the second stage (**Algorithm 3 FIND-TOPO**), r uses the IRV information obtained to recover the topology. An interesting feature of the algorithms proposed is that random network failures actually make it *easier* to efficiently estimate the topology.

Assumptions, justifications, and notation:

1) *Multiple "successful" source generations.* A "successful" generation means the number of errors does not exceed the bound $C - 1$ and receiver r can decode the source message correctly by using network error-correcting-codes (see Section II-G for details). The protocol runs for t independent "successful" source generations, where t is a design parameter chosen to trade off between the probability of success and the computational complexity of the topology estimation protocol. Let $X(i)$ be the source messages transmitted, $\mathcal{Z}(i)$ be set of edges suffering errors and $Y(i)$ be the received matrix in the ith source generation.
2) *Weak connectivity requirement.* It is assumed that each internal node has out-degree no less than 2. Note that this is a necessary condition to ensure that each edge is *distinguishable* from every other edge, *i.e.*, any pair of edges are flow-independent (see the definition in Section II-C for details).
3) *Each node knows the IDs of its neighbors.* As in Section IV-B, Assumption 3.
4) *The network is not "noodle like".* That is, the network does not have high-depth but narrow-width[5]. To be precise for any distinct $i, j \in [1, t]$ let the random variable $\mathcal{D}(i, j)$ be 1 if and only if $\mathcal{Z}(i)$ is flow-independent to $\mathcal{Z}(j)$, *i.e.*, *flow-rank*$(\mathcal{Z}(i) \cup \mathcal{Z}(j)) =$ *flow-rank*$(\mathcal{Z}(i))$ + *flow-rank*$(\mathcal{Z}(j))$. Since random network errors are independent of the source generation, $Pr(\mathcal{D}(i, j) \neq 1)$ has no dependence on (i, j) and is defined as p_c. The property that the network not be "noodle-like" requires that p_c be bounded away from 1.
5) *Independent errors.* For each source generation, each edge e independently has random errors with probability

[5]At a high-level, the problem lies in the fact that such networks have high description complexity (dominated by the height), but can only support a low information rate (dominated by the width).

at least p. Note that Assumptions 1 and 4 above require that the typical number of error edges $p|\mathcal{E}|$ in each source generation is no more than C. Thus we can assume $p = \Theta(1/|\mathcal{E}|)$.

6) *Weak type common randomness is assumed.* This assumption is justified by Theorem 3.

Stage I: Find candidate IRVs

In the i'th source generation, the source message is a matrix $X(i)$ in $\mathbb{F}_q^{C\times n}$, where the first C columns of $X(i)$ form an $C \times C$ identity matrix and the last $n - C$ columns of $X(i)$ are the real messages. For any matrix N with n columns, let N_h (and N_m) be the matrix comprised of the first C columns (and last $n-C$ columns) of N. Then the algorithm that finds a set of candidate IRVs is as follows.

Algorithm 2 FIND-IRV

Input: $\{Y(i), i \in [1, t]\}$

Output: $\mathcal{I}_{IRV}$, which is a set of dimension-one subspaces in $\mathbb{F}_q^C$ and initialized as an empty set

1: **for** $i = 1$ to t **do**
2: Compute $X(i)$ using network error-correction-codes
3: Compute $E(i)_r = Y(i)_m - Y(i)_h X(i)_m$
4: **end for**
5: **for** $i, j = 1$ to t and $i \neq j$ **do**
6: Compute the intersection of the column-spaces $\mathbf{E(i)_r} \cap \mathbf{E(j)_r}$
7: **if** rank$(\mathbf{E(i)_r} \cap \mathbf{E(j)_r}) = 1$ **then**
8: Add $\mathbf{E(i)_r} \cap \mathbf{E(j)_r}$ into $\mathcal{I}_{IRV}$
9: **end if**
10: **end for**

Let p_a denote $p_c + 2p_s + |\mathcal{E}|/q$ and let p_s denote $1 - (1 - z/q)[1 - 2C^2/(n - C)]$ and let $< \mathbf{v} >$ denote the one-dimensional subspace spanned by the vector $\mathbf{v}$. Then the following theorem characterizes the performance of **FIND-IRV**.

Theorem 8: The probability that $\mathcal{I}_{IRV}$ contains $\{< \mathbf{t}'(e) >: e \in \mathcal{E}\}$ is at least $1 - |\mathcal{E}|p_a^{tp/2}$.

The proof is presented after the discussion and Lemma 9 below. Note that the probability p_s asymptotically approaches 0 with increasing block-length n and field-size q. Hence p_a is bounded away from 1 using Assumption 4. Thus if $t = \Theta(\log(|\mathcal{E}|)/p)$, the probability that $\mathcal{I}_{IRV}$ contains $\{< \mathbf{t}'(e) >: e \in \mathcal{E}\}$ is $1 - o(1)$. Since $p = \Theta(1/|\mathcal{E}|)$, without loss of generality we henceforth assume $t = \Theta(|\mathcal{E}| \log(|\mathcal{E}|))$ in future sections.

We note that at this stage elements in $\mathcal{I}_{IRV}$ has no correspondence with edges in the network– such correspondences shall we found in the next stage by the algorithm **FIND-TOPO** later. Furthermore, the set of vectors output by **FIND-IRV** can also include some "fake candidates", as demonstrated in the example in Figure 9. In the next stage of topology estimation, these fake IRVs will be automatically filtered out by **FIND-TOPO**.

As a precursor to proving Theorem 8, we first characterize the set of IRVs arising from random errors via Lemma 9 below.

Lemma 9: Under the random error model, with probability at least $1 - p_s$, $\mathbf{E(i)_r} = \mathbf{T}'(\mathcal{Z}(\mathbf{i}))$.

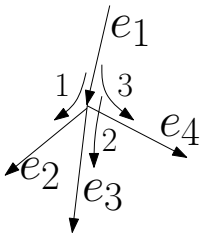


Fig. 9. Let $\mathcal{Z}(1) = \{e_1, e_4\}$ and $\mathcal{Z}(2) = \{e_2, e_3\}$ and $rank(\mathbf{t}'(e_2), \mathbf{t}'(e_3), \mathbf{t}'(e_4)) = 3$, then we have $rank(\mathbf{T}'(\mathbf{e_2}, \mathbf{e_3}) \cap \mathbf{T}'(\mathbf{e_1}, \mathbf{e_4})) = 1$ and $\mathbf{T}'(\mathbf{e_2}, \mathbf{e_3}) \cap \mathbf{T}'(\mathbf{e_1}, \mathbf{e_4}) = [\mathbf{t}'(\mathbf{e_2}) + \mathbf{2t}'(\mathbf{e_3})]$, which is a "fake candidate".

Proof: Recall that $Z(i)_m$ comprises of the last $n-C$ columns of Z. We first prove that $Z(i)_m$ has full row rank z with probability at least $1 - p_s$.

In the random error model (see Section II-E for details) each error edge e has at least one randomly chosen location (say ℓ) in the injected packet $\mathbf{z}(e)$ such that the ℓth component of $\mathbf{z}(e)$ is chosen uniformly at random from $\mathbb{F}_q$. Thus for each row of $Z(i)$, all the last $n - C$ elements are zero with probability at most C/n. Using the Union Bound [33] $Z(i)_m$ has zero rows with probability at most C^2/n. Thus in the following we assume each row of $Z(i)_m$ is non-zero.

The "Birthday Paradox" [33] implies that with probability at least $1-C^2/(n-C)$, for each row of $Z(i)_m$, the following happens: there are z *distinct* column indexes $l_1, \ldots, l_z \in \{1, \ldots, n - C\}$ such that $Z(i)_m(i, l_i)$ is chosen uniformly at random. Then the determinant of the sub-matrix of the $\{l_1, \ldots, l_z\}$th columns of $Z(i)_m$ is a nonzero polynomial of degree z of uniformly random variables over $\mathbb{F}_q$. By the Schwartz-Zippel Lemma [32] this determinant is non-zero with probability at least $(1 - z/q)$. Thus $Z(i)_m$ has z independent columns with probability at least $(1 - z/q)[1 - 2C^2/(n - C)] = 1 - p_s$.

From the definition we have $Y(i)_m = TX(i)_m + T'(\mathcal{Z}(i))Z(i)_m$ and $Y(i)_h = T + T'(\mathcal{Z}(i))Z(i)_h$. Hence, we have $E(i)_r = T'(\mathcal{Z}(i))(Z(i)_m - Z(i)_h X(i)_m)$. Since the non-zero random variables in $Z(i)_m$ are chosen independently from $Z(i)_h X(i)_m$, the matrix$(Z(i)_m - Z(i)_h X(i)_m)$ has full row rank z with the same probability $1 - p_s$. Thus $\mathbf{E(i)_r} = \mathbf{T}'(\mathcal{Z}(\mathbf{i}))$ with probability at least $1 - p_s$. □

Then we have:

Proof of Theorem 8: For any edge e and any $i \neq j \in \{1, \ldots, t\}$ and $e \in \mathcal{Z}(i) \cap \mathcal{Z}(j)$, we compute the probability of the event $\mathcal{F}(e, i, j)$, which is defined as the one-dimensional subspace $< \mathbf{t}'(\mathbf{e}) >$ equaling the subspace $\mathbf{E(i)_r} \cap \mathbf{E(j)_r}$.

By Assumption 4, with probability at least $1 - p_c$, set $\mathcal{Z}(i) - e$ is flow-independent of $\mathcal{Z}(j) - e$. Conditioned on this, Lemma 1.2 implies that with probability at least $1-p_c-|\mathcal{E}|/q$, $\mathbf{T}'(\mathcal{Z}(\mathbf{i})\backslash\mathbf{e})$ is linearly independent of $\mathbf{T}'(\mathcal{Z}(\mathbf{j})\backslash\mathbf{e})$. Hence $\mathbf{T}'(\mathcal{Z}(\mathbf{i})) \cap \mathbf{T}'(\mathcal{Z}(\mathbf{j}))$ equals $< \mathbf{t}'(\mathbf{e}) >$. And by Lemma 9, either of $\mathbf{E(i)_r} \neq \mathbf{T}'(\mathcal{Z}(\mathbf{i}))$ or $\mathbf{E(j)_r} \neq \mathbf{T}'(\mathcal{Z}(\mathbf{j}))$ with probability at most p_s. Conditioning on all the events implies that the probability of event $\mathcal{F}(e, i, j)$ is at least $1-p_c-2p_s-|\mathcal{E}|/q$.

When t is large enough, by the Chernoff bound [33] e will fail at least $tp/2$ times with probability at least $1 - p^{\Theta(t)}$. Conditioned on these many failures, there are $tp/4$ proba-

bilistically independent $\mathcal{F}(e,i,j)$ for edge e, and **FIND-IRV** accepts $\mathbf{t}'(e)$ with probability at least $1-(p_a^{tp/4}+p^{\Theta(t)})$. Taking the union bound over all edges gives the required result. □

Stage II: Topology recovery via candidate IRVs

Using $\mathcal{I}_{IRV}$, we now describe **Algorithm 3 FIND-TOPO** that determines the network topology.

Note that $\mathcal{I}_{IRV}$ is merely a set of one-dimensional subspaces, and as such, individual element may have no correspondence with the actual IRV of any edge in the network. At any point in **FIND-TOPO**, let $\bar{\mathcal{G}}$ denote the network topology recovered thus far. Let $\bar{\mathcal{V}}$ and $\bar{\mathcal{E}}$ be the corresponding sets of nodes and edges respectively in $\bar{\mathcal{G}}$, and $\bar{\mathcal{I}}_{IRV}$ be the set of IRVs of the edges in $\bar{\mathcal{E}}$, which are computed from $\bar{\mathcal{G}}$ and the set of local random code-books $\mathcal{R}=\{\mathcal{R}_v : v\in\mathcal{V}\}$. We note that the IRVs in $\bar{\mathcal{I}}_{IRV}$ are vectors rather than one-dimensional subspaces.

Algorithm 3 FIND-TOPO

Input: $\mathcal{I}_{IRV}$ and codebook $\mathcal{R}$
Output: A output graph $\bar{\mathcal{G}}=(\bar{\mathcal{V}},\bar{\mathcal{E}})$

1: The set $\bar{\mathcal{V}}$ is initialized as the receiver r, all its upstream neighbors, and the source s
2: The set $\bar{\mathcal{E}}$ is initialized as the set of edges incoming to r
3: The set of $\bar{\mathcal{I}}_{IRV}$ is initialized as the IRVs of the incoming edges of r, *i.e.*, a subset of distinct columns of the $C\times C$ identity matrix
4: NewEdge←**true**
5: **while** NewEdge=**true do**
6: NewEdge←**false**
7: **for** $v\in\bar{\mathcal{V}}$ and $v\neq s$ **do**
8: Assume $e_1,\ldots,e_d$ be the outgoing edges of v in $\bar{\mathcal{G}}$
9: **if** $\{\bar{\mathbf{t}}'(\mathbf{e_1}),\ldots,\bar{\mathbf{t}}'(\mathbf{e_d})\}$ from $\bar{\mathcal{I}}_{IRV}$ has rank larger than 1 **then**
10: **for** Each candidate edge $e(u,v)$, $e(u,v)\notin\bar{\mathcal{E}}$ **do**
11: Use $\mathcal{R}$ to compute the IRV of e as $\bar{\mathbf{t}}'(\mathbf{e})=\sum_{j=1}^{d}\beta(e,v,e_j)\bar{\mathbf{t}}'(\mathbf{e_j})$
12: **if** $<\bar{\mathbf{t}}'(\mathbf{e})>\in\bar{\mathcal{I}}_{IRV}$ **then**
13: NewEdge←**true**
14: If $u\notin\bar{\mathcal{V}}$, add u to $\bar{\mathcal{V}}$
15: Add $e=e(u,v)$ to $\bar{\mathcal{E}}$
16: Based on $\mathcal{R}$, update $\bar{\mathcal{I}}_{IRV}$ from $\bar{\mathcal{G}}=(\bar{\mathcal{V}},\bar{\mathcal{E}})$
17: **end if**
18: **end for**
19: **end if**
20: **end for**
21: **end while**

If $\mathcal{I}_{IRV}$ contains all edge IRVs as claimed by Theorem 8, we show correctness of **FIND-TOPO** as follows.

Theorem 10: With probability $1-\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^4|\mathcal{V}|)/q$, **FIND-TOPO** recovers the network topology accurately by performing $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^4|\mathcal{V}|C)$ operations over $\mathbb{F}_q$.

Before the proof of Theorem 10 we need the following lemma, which states that with high probability the algorithm accepts an edge e as a correct edge if and only if e is actually in the network $\mathcal{G}$.

Lemma 11: 1) If edge $e=(u,v)$ exists in $\mathcal{G}$, $<\mathbf{t}'(e)>$ is in $\mathcal{I}_{IRV}$, $\{e_1,\ldots,e_d\}$ are exactly all the outgoing edges of v in $\mathcal{G}$ and $\bar{\mathbf{t}}'(\mathbf{e_i})=\mathbf{t}'(e_i)$ for $i=1,2,...,d$, the algorithm accepts e with probability 1.
2) If edge e does not exist in $\mathcal{G}$, the algorithm accepts e with probability at most $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^2)/q$.

Proof:

1) By the definition of IRV we have $\bar{\mathbf{t}}'(\mathbf{e})=\sum_{j=1}^{d}\beta(e,v,e_j)\bar{\mathbf{t}}'(\mathbf{e_j})=\mathbf{t}'(e)$ and will be accepted.
2) If e does not exist in $\mathcal{G}$, the coding coefficients $\{\beta(e,v,e_j) : j=1,\ldots,d\}$ are not used. Hence from the perspective of any element $<\mathbf{h}>$ in $\mathcal{I}_{IRV}$, $\sum_{j=1}^{d}\beta(e,v,e_j)\bar{\mathbf{t}}'(\mathbf{e_j})$ is an independently and uniformly chosen vector in the span of the vectors $\{\bar{\mathbf{t}}'(\mathbf{e_j})\,:\,\mathbf{j}\in\{\mathbf{1},\ldots,\mathbf{d}\}\}$. Since line 12 is called only if the rank of $\{\bar{\mathbf{t}}'(\mathbf{e_j})\,:\,\mathbf{j}\in\{\mathbf{1},\ldots,\mathbf{d}\}\}$ is no less than 2, $\bar{\mathbf{t}}'(\mathbf{e})\in<\mathbf{h}>$ with probability at most $1/q$. Since **FIND-IRV** in Stage I needs at most $t=\mathcal{O}(\log(|\mathcal{E}|)|\mathcal{E}|)$ source generations[6], $\mathcal{I}_{IRV}$ has size at most $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^2)$. Using the union bound [33], $<\bar{\mathbf{t}}'(\mathbf{e}(\mathbf{u},\mathbf{v},\mathbf{i}))>$ is in $\mathcal{I}_{IRV}$ with probability at most $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^2/q)$. □

Then we have:

Proof of Theorem 10: Note that if no error occurs, the algorithm can find at most $|\mathcal{E}|$ edges, the task of finding a new edge requires at most $|\mathcal{V}|$ invocations of line 7 (once for each node), and each invocation of line 7 results in at most $|\mathcal{E}|$ invocations of line 10. Thus line 10 can be invoked at most $|\mathcal{E}|^2|\mathcal{V}|$ times. Lemma 11 demonstrates that each invocation results in an error with probability at most $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^2/q)$. Note further that this is the only possible error event. Hence by the union bound [33], the probability that **FIND-TOPO** results in an erroneous reconstruction of $\mathcal{G}$ is $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^4|\mathcal{V}|)/q$.

As shown in the proof of Lemma 11 above, there are at most $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^2)$ elements in $\mathcal{I}_{IRV}$. For each element in $\mathcal{I}_{IRV}$, it takes $\mathcal{O}(C)$ field operations to decide whether it equals the target one-dimensional space $<\bar{\mathbf{t}}'(\mathbf{e})>$. Thus, each computation of Line 12 takes at most $\mathcal{O}(\log^2(|\mathcal{E}|)|\mathcal{E}|^2C)$ finite field comparisons to determine membership of $<\bar{\mathbf{t}}'(\mathbf{e})>$ in $\mathcal{I}_{IRV}$. Hence, given that the bound on the number of invocations of line 12 and that this can be verified to be the most computationally expensive step, the running-time of **FIND-TOPO** is $O(\log^2(|\mathcal{E}|)|\mathcal{E}|^4|\mathcal{V}|C)$ operations over $\mathbb{F}_q$.

Finally, we note that $\mathcal{G}$ is acyclic and the assumption that $\mathcal{I}_{IRV}$ contains $\{<\mathbf{t}'(e)>: e\in\mathcal{E}\}$. Hence conditioning on no incorrect edge being accepted, for each invocation of line 5, unless $\bar{\mathcal{G}}=\mathcal{G}$, there exists an edge e such that all edges e' downstream of e in $\mathcal{G}$ are in $\bar{\mathcal{E}}$, which implies all the corresponding $\bar{\mathbf{t}}'(\mathbf{e}')$s are correctly computed. Thus by Lemma 11 edge e is accepted into $\bar{\mathcal{E}}$ in line 15 with probability 1. Hence, each edge actually in $\mathcal{G}$ also eventually ends up in $\bar{\mathcal{G}}$, and **FIND-TOPO** terminates. □

V. Error localization for RLNC

As in previous works ([9], [15], [16]), under RLNC the receiver r must know the network topology and local random coding coefficients to locate network errors. Thus in this

[6]As pointed out in Remark 2 after Theorem 8.

section receiver r is assumed to know the IRV of each edge, which can be from the topology estimation algorithms in Section IV, or as a part of priori network design.

A. Locating adversarial errors under RLNC

In this subsection we demonstrate how to detect the network edges where the adversary injects errors. Since the IRV is the fingerprint of the corresponding edge, detecting the edges where errors have been injected becomes mathematically equivalent to the problem of detecting IRVs in the error matrix E. Our technique is based on the fact that when the edges are flow-independent (see the definition in Section II-C for details) enough from each other, the IRV of each erroneous edge is not erasable from the column space of the error matrix E.

Assumptions and justifications:

1) Each internal node has out-degree at least $2z$. Since $\mathcal{G}$ is acyclic, this implies that every set of $2z$ edges in $\mathcal{G}$ is flow-independent. While this assumption seems strong, we demonstrate in Theorem 13 that such a condition is necessary for r to identify the locations of z corrupted edges.
2) At most z edges in $\mathcal{Z}$ suffer errors, *i.e.*, $\{e : e \in \mathcal{E}, \mathbf{z}(e) \neq 0\} = \mathcal{Z}$ and $|\mathcal{Z}| \leq z$. When $2z + 1 \leq C$, network-error-correcting codes (see Section II-G for details) are used so that the source message X (and thus the error matrix E) is provably decodable.

In the following, we present the algorithm to locate the network adversaries under RLNC.

Algorithm 4 LOCATE-ADVERSARY-RLNC

Input: Matrix E and IRVs $\{\mathbf{t}'(e) : e \in \mathcal{E}\}$

1: Compute rank(E)$= \eta$
2: Let $\{\mathbf{e_1}, \mathbf{e_2}, ..., \mathbf{e}_\eta\}$ be a set of independent columns of E
3: **for** $i = 1$ to η **do**
4: Find a set of edges $\mathcal{Z}_i$ with minimal cardinality such that $\mathbf{e_i}$ is in the column space of the corresponding impulse response matrix $T'(\mathcal{Z}_i)$
5: **end for**
6: **return** $\mathcal{Z}' = \cup_{i\in[1,\eta]}\mathcal{Z}(i)$

We show that with high probability **LOCATE-ADVERSARY-RLNC** finds the location of edges with adversarial errors.

Theorem 12: With probability at least $1 - |\mathcal{E}|\binom{|\mathcal{E}|}{2z}/q$ the solution of **LOCATE-ADVERSARY-RLNC** results in $\mathcal{Z}' = \mathcal{Z}$.

Proof: Note that Assumption 1, with high probability, gives a similar statement about the rank of the corresponding IRVs. Using the Union Bound [33] on the result of Lemma 1.2 gives us the result that any $2z$ IRVs are independent with probability at least $1 - |\mathcal{E}|\binom{|\mathcal{E}|}{2z}/q$. We henceforth assume this happens in the following.

First of all, since each $\mathbf{e_i}$ is in $\mathbf{T}'(\mathcal{Z})$, we have $|\mathcal{Z}_i| \leq z$ for each $i = 1, 2, ..., \eta$.

We claim that for each $i \in \{1, 2, \ldots, \eta\}$, $\mathcal{Z}_i$ must be a subset of $\mathcal{Z}$. If not, say $e \in \mathcal{Z}_i$ is not in $\mathcal{Z}$. By the definition of **LOCATE-ADVERSARY-RLNC**, $\mathbf{t}'(e)$ is in the span of the columns of $T'(\mathcal{Z})$ and $T'(\mathcal{Z}_i - e)$. Thus a non-trivial combination of the at most $2z - 1$ IRVs results in $\mathbf{t}'(e)$. It contradicts the assumption that any $2z$ IRVs are linearly independent.

We prove next that for any edge $e \in \mathcal{Z}$ on which the adversary injects a non-zero error, **LOCATE-ADVERSARY-RLNC** outputs at least one $\mathcal{Z}_i$ such that $e \in \mathcal{Z}_i$. Without loss of generality, let e be the first edge in $\mathcal{Z}$. Then $E = T'(\mathcal{Z})Z$ and the first row of Z is nonzero. Since any z IRVs are independent, $T'(\mathcal{Z})$ is of full column rank. Then for any η independent columns in E there must be at least one, say $\mathbf{e_i}$, such that the IRV $\mathbf{t}'(e)$ has nonzero contribution to it. That is, $\mathbf{e_i} = T'(\mathcal{Z})(c_1, c_2, ..., c_z)^T$ with $c_1 \neq 0$. Hence running **LOCATE-ADVERSARY-RLNC** on $\mathbf{e_i}$ will find $\mathbf{t}'(e)$ and include the corresponding edge e into $\mathcal{Z}_i$. Otherwise, $\mathbf{t}'(e)$ is in the span of the columns of $T'(\mathcal{Z} - e)$ and $T'(\mathcal{Z}_i)$, which contradicts the assumption that any $2z$ IRVs are linearly independent. □

We now show matching converses for Theorem 12. In particular, we demonstrate in Theorem 13 that Assumption 1 (*i.e.*, that any $2z$ edges are flow-independent) is necessary.

Theorem 13: For linear network coding, any z corrupted edges are detectable if and only if any $2z$ edges are flow-independent.

Proof: The "if" direction is a corollary of Theorem 12. For the "only if" direction, suppose there exist $2z$ edges such that they are not flow-independent. Then the corresponding IRVs cannot be linearly independent by Lemma 1.1. Then there must exist a partition of these $2z$ edges into two edge sets $\mathcal{Z}_1$ and $\mathcal{Z}_2$ such that $|\mathcal{Z}_1| = z$ and $|\mathcal{Z}_2| = z$ and $\mathbf{T}'(\mathcal{Z}_1) \cap \mathbf{T}'(\mathcal{Z}_2) \neq \{0\}$, *i.e.,* the *spanning spaces* of the corresponding IRVs in the two sets intersect non-trivially. Then the adversary can choose to corrupt $\mathcal{Z}_1$ and inject errors Z in a manner such that the columns of $T'(\mathcal{Z}_1)Z$ are in $\mathbf{T}'(\mathcal{Z}_2)$. This means r cannot distinguish whether the errors are from $\mathcal{Z}_1$ or $\mathcal{Z}_2$. □

Theorem 13 deals with the case that any z edges can be corrupted. If only some sets of edges are candidates for adversarial action (for instance the set of outgoing edges from some "vulnerable" nodes) we obtain the following corollary.

Corollary 14: Let $\mathcal{S} = \{\mathcal{Z}_1, \mathcal{Z}_2, ..., \mathcal{Z}_t\}$ be disjoint sets of edges such that exactly one of them is controlled by an adversary. Then r can detect which edge set is controlled by the adversary if and only if any two sets $\mathcal{Z}_i$ and $\mathcal{Z}_j$ in $\mathcal{S}$ are flow-independent.

Note: The flow-independence between edge-sets $\mathcal{Z}_i$ and $\mathcal{Z}_j$ in $\mathcal{S}$ does not require the edges within either of $\mathcal{Z}_i$ or $\mathcal{Z}_j$ to be flow-independent. It merely requires that flow-rank($\mathcal{Z}_i$) + flow-rank($\mathcal{Z}_j$) = flow-rank($\mathcal{Z}_i \cup \mathcal{Z}_j$).

Note that running **LOCATE-ADVERSARY-RLNC** might require checking all the $\binom{\mathcal{E}}{z}$ subsets of edges in the network – this is exponential in z. We now demonstrate that for networks performing RLNC, the task of locating the set of adversarial edges is in fact computationally intractable even when the receiver knows the topology and local encoding coefficients in advance.

Theorem 15: For RLNC, if knowing the network $\mathcal{G}$ and all local coding coefficients allows the receiver r correctly

locating all adversarial locations in time polynomial in network parameters, NCPRLC (see the definition in Section II-H for details) can be solved in time polynomial in problem parameters.

Proof: Given a NCPRLC instance $(H, z, \mathbf{e})$, as shown in Figure 8, we construct a network with l_1 edges to receiver r and l_2 edges to node u.

Since H is a matrix chosen uniformly at random over $\mathbb{F}_q$, it corresponds to a RLNC, where each column of H corresponds to an IRV of an incoming edge of u.

Assume the adversary corrupts z incoming edges of u. The adversary can choose the errors Z such that each column of $E = T'(\mathcal{Z})Z$ equals $\mathbf{e}$. Also, E is all the information about the adversarial behavior known by r under RLNC. Any algorithm that outputs the corrupted set $\mathcal{Z}$ must satisfy $\mathbf{e} \in \mathbf{T}'(\mathcal{Z})$ and $|\mathcal{Z}| \leq z$. Once $\mathcal{Z}$ is found, r actually solves the NCPRLC instance $(H, z, \mathbf{e})$. □

B. Locating random errors under RLNC

In the previous section, we considered the problem of locating adversarial edge errors over random linear network coding, and proved the computational hardness of the problem. In this section, we consider random edge errors, as defined in Section II-E. We propose a different error locating algorithm algorithm whose time-complexity is polynomial in network parameters. In particular, this new algorithm (denoted LOCATE-RANDOM-RLNC) does not require that the graph has high connectivity (as we demonstrated in Theorem 13 is a necessary condition for locating adversarial edge errors).

Since $\mathbf{T}'(\mathcal{Z}) = \mathbf{T}'(Ext(\mathcal{Z}))$ (see the definition of $Ext(\mathcal{Z})$ in Section III-A for reference), the receiver cannot distinguish whether the errors are from $\mathcal{Z}$ or $Ext(\mathcal{Z})$. So rather than finding $\mathcal{Z}$, we provide a computationally tractable algorithm to locate $Ext(\mathcal{Z})$, a proxy for $\mathcal{Z}$. The algorithm that finds $Ext(\mathcal{Z})$ is as follows. Recall that for any matrix N with n columns, N_h (and N_m) be the matrix comprising of the first C columns (and last $n - C$) of N.

Algorithm 5 LOCATE-RANDOM-RLNC

Input: Matrix Y and IRVs $\{\mathbf{t}'(e) : e \in \mathcal{E}\}$

1: $\mathcal{Z}' \leftarrow \emptyset$
2: Compute the source message X using network error-correction codes
3: Compute $E_r = Y_m - Y_h X_m$
4: **for** Each edge $e \in \mathcal{E}$ **do**
5: **if** IRV $\mathbf{t}'(e)$ lies in $\mathbf{E_r}$ **then**
6: Add e into $\mathcal{Z}'$
7: **end if**
8: **end for**
9: **return** $\mathcal{Z}'$

The correctness of **LOCATE-RANDOM-RLNC** is proved as follows.

Theorem 16: If z is no more than $C - 1$, $\mathcal{Z}' = Ext(\mathcal{Z})$ with probability at least $1 - 3|\mathcal{E}|^2/q - 2C^2/(n - C)$. The computational complexity is $\mathcal{O}(|\mathcal{E}|C^2)$ operations over $\mathbb{F}_q$.

Proof: Lemma 1.2 implies that with probability at least $1 - 2|\mathcal{E}|/q$, $\mathbf{T}'(\mathcal{Z})) = \mathbf{T}'(Ext(\mathcal{Z}))$. Lemma 9 implies that with probability at least $1 - 2C^2/(n - C)$, $\mathbf{E_r} = \mathbf{T}'(\mathcal{Z}))$. Thus, using Union Bound, with probability at least $1 - 2|\mathcal{E}|/q - 2C^2/(n - C)$, we have $\mathbf{E_r} = \mathbf{T}'(Ext(\mathcal{Z}))$. Thus we have $Ext(\mathcal{Z}) \subseteq \mathcal{Z}'$.

For the other direction, using the Union Bound over all $|\mathcal{E}|$ edges on Lemma 1.2, with probability at least $1 - |\mathcal{E}|^2/q$, for any edge $e \notin Ext(\mathcal{Z})$, $\mathbf{t}'(e)$ is not in $\mathbf{E_r}$. Therefore, $\mathcal{Z}' \subseteq Ext(\mathcal{Z})$.

Combining the two, we have that $Ext(\mathcal{Z}) = \mathcal{Z}'$ with probability at least $1 - 3|\mathcal{E}|^2/q - 2C^2/(n - C)$.

For each IRV $\mathbf{t}'(e)$, it costs at most $\mathcal{O}(C^2)$ operations over $\mathbb{F}_q$ to check whether it is in $\mathbf{E_r}$. Then the total computational complexity of **LOCATE-RANDOM-RLNC** is $\mathcal{O}(|\mathcal{E}|C^2)$ operations over $\mathbb{F}_q$. □

VI. Network Reed-Solomon Coding (NRSC)

In the second part of the paper, we construct a particular type of RLNC, network Reed-Solomon coding (NRSC). This part consists of Sections VI, Section VII and Section VIII, wherein we respectively define NRSC, and show its application to the location of adversarial edges in networks, and that of topology estimation in the presence of random errors.

A. Motivations

In part I (Sections IV and V), under random linear network coding (RLNC), network tomography is studied for both adversarial and random error models. For the random error model the schemes for both topology estimation and error localization can be done in time polynomial in network parameters, while the schemes presented for the adversarial model all require time exponential in network parameters. Moreover, under RLNC localizing adversarial errors is computationally intractable (see Theorem 15) and requires the knowledge of network topology, for which the estimation algorithm we present also requires time exponential in network parameters.

In this section network Reed-Solomon Coding (NRSC) is proposed to improve the tomographic performance (specially for the adversarial error model), while preserving the key advantages of RLNC. To be concrete, NRSC has the following features.

1) *Low implementation complexity*. The proposed NRSC is a linear network coding scheme (see Section II-D for details), and can be implemented in a *distributed and efficient* manner, where each network node only needs to know the node-IDs of its adjacent neighbors. Thus once an edge (or node) has left or joined, only its adjacent neighbors need to adjust the coding coefficients.

2) *High throughput*. The multicast capacity of the underlying communication scenario is achieved with high probability.

3) NRSC aids tomography in the following two aspects. i) *Computational efficiency*. Under the adversarial error model, the receiver under NRSC can locate a number of adversarial errors that match a corresponding tomographic upper bound (see Theorem 13 for details) in a computationally efficient manner. For the random error model, a lightweight topology

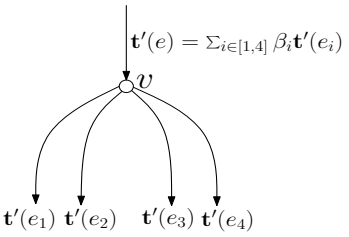


Fig. 10. The IRV of e is a linear combination of the IRVs of e_1, e_2, e_3 and e_4.

estimation algorithm is provided under NRSC. ii) *Robustness for dynamic networks*. For adversarial (and random) error localization, the algorithms we present under NRSC do not require *a priori* knowledge of the network topology and thus are robust against edge and node updates. [7] Hence, for topology estimation in the random error model, the lightweight algorithm under NRSC fits dynamic networks better than the one under RLNC, since the latter requires that the network topology remain stable for a long time period.

B. Overview of NRSC

In NRSC, in addition to an IRV each edge e is also assigned a *virtual IRV* (VIRV) $\mathbf{t}''(e)$. This virtual IRV is a deterministic function of the node-IDs of the head and tail of e (and hence is known to them), and must be a column of the parity check matrix of a point-to-point Reed Soloman (RS) code [29]. Further, each node in an NRSC (say node v in Figure 10) carefully chooses its coding coefficients (*e.g.*, $\{\beta_1, ..., \beta_4\}$ at node v in Figure 10, where $\beta_i = \beta(e, v, e_i)$ for $i = 1, ..., 4$) such that the virtual IRVs of edges entering and leaving v satisfy the same linear relationship as the IRVs (in the case of Figure 10, $\mathbf{t}''(e) = \beta_1\mathbf{t}''(e_1) + ... + \beta_4\mathbf{t}''(e_4)$). In other words, under NRSC every network node makes a "local contribution" to force edge IRVs to equal the corresponding VIRVs. And this objective can be achieved if and only if a connectivity requirement is satisfied (see Corollary 22 for details).

At a high level, we present the ideas that help NRSC improve upon RLNC in terms of the performance of the tomographic algorithms.

1) *Computational efficiency*. Under RLNC, each edge IRV is randomly chosen from the linear subspace spanned by the down-stream edge IRVs, resulting in network codes for which locating network adversaries is as hard as NCPRLC (see Theorem 15 for details). Under NRSC, VIRVs (and then IRVs) are smartly chosen so as to enable computationally efficient adversarial location.

2) *Robustness for dynamic networks*. Under RLNC each edge updates (say, disconnecting) results in IRV updates for all upstream edges. Under NRSC, once an edge is disconnected, its tail node adjusts its local coding coefficients to ensure that the IRVs remain unchanged for all upstream nodes. This feature significantly reduces the complexity of error localization over dynamic networks.

[7]Note that under RLNC, the error localization algorithms in previous works [9], [14], [16] and this paper require prior knowledge of the network topology. However, as highlighted in Section IV-B, topology estimation under RLNC for networks with adversarial errors is computationally intractable (though Section V-B presents a computationally tractable algorithm for networks with random errors).

C. Node and edge IDs

Each pair of nodes (u, v) in $\mathcal{V} \otimes \mathcal{V}$ has an ID $id(u, v)$ chosen independently and uniformly at random from $\mathbb{F}_q$.[8] These IDs can be a part of network design, or outputted by a pseudorandom hash function (with input as a pair of nodes) such as AES [35] that can be accessed by all parties. Thus this set of $|\mathcal{V}|^2$ IDs is publicly known *a priori* to all parties (including the adversaries), even though they may not know *which* nodes and edges are actually in the network.

The following lemma shows that each node pair has a distinct ID with high probability:

Lemma 17: With probability at least $1 - |\mathcal{V}|^4/q$, for any $(u, v) \neq (u', v')$ in $\mathcal{E}$, $id(u, v) \neq id(u', v')$.

Proof: For any $(u, v) \neq (u', v')$, $id(u, v) = id(u', v')$ with probability at most $1/q$. Since $\mathcal{V} \times \mathcal{V}$ has size $|\mathcal{V}|^2$, there are at most $\binom{|\mathcal{V}|^2}{2} < |\mathcal{V}|^4$ distinct pairs in $\mathcal{V} \times \mathcal{V}$. Using union bound [33] over all these pairs, the lemma is true with probability at least $1 - |\mathcal{V}|^4/q$. □

For each edge $e(u, v) \in \mathcal{E}$ the ID of e is $id(e) = id(u, v)$. Thus the ID of edge $e(u, v)$ can be figured out by both u and v if they know their adjacent neighbors. A direct corollary of Lemma 17 is that each edge has a distinct ID with high probability. We henceforth assume that this is indeed the case.

For each edge e the *virtual impulse response vector* (VIRV) is $\mathbf{t}''(e) \in \mathbb{F}_q^C$, which is $[id(e), (id(e))^2, ..., (id(e))^C]^T$. For any set of edges $\mathcal{Z}$ with size z, the virtual impulse-response-matrix (VIRM) is $T''(\mathcal{Z}) \in \mathbb{F}_q^{C\times z}$, with the columns comprised of $\{\mathbf{t}''(e), e \in \mathcal{Z}\}$.

For the ease of notation we also defined a dimension-parameterized VIRV as $\mathbf{t}''(e, i) = [id(e), (id(e))^2, ..., (id(e))^i]^T$. For any set of edges $\mathcal{Z}$ with size z, the corresponding VIRM is $T''(\mathcal{Z}, i) \in \mathbb{F}_q^{i\times z}$, with the columns comprised of $\{\mathbf{t}''(e, i), e \in \mathcal{Z}\}$. Note that $T''(\mathcal{Z}, z)$ is a Vandermonde matrix and invertible when $|\mathcal{Z}| = z$ and the edges in $\mathcal{Z}$ have distinct IDs.

D. Code construction of NRSC

We assume by default that the edges in $\mathcal{E}$ have distinct IDs, which happens with probability at least $1 - |\mathcal{V}|^4/q$ by Lemma 17. Recall that C is the capacity of the network, *i.e.*, $C = \text{max-flow}(s, r)$, and for ease of notation we assume that the source has exactly C outgoing edges and the receiver has C incoming edges (see Section II-B for details).

The construction of NRSC is then as follows.

Source encoder: Let $\mathbf{Out}(s) = \{e_1, e_2, ..., e_C\}$ be the outgoing edges of the source s and $X \in \mathbb{F}_q^{C\times n}$ be the source message matrix. The source s computes $M = T''(\mathbf{Out}(s), C)^{-1}X$ and sends the ith row of M as the packet over e_i. Note that X contains a known "header", say the $C \times C$ identity matrix over $\mathbb{F}_q$, to indicate the network transform to the receiver.

[8]Note that for scenario where parallel edges are allowed, we assume some pairs of nodes have multiple IDs, the i'th of which is the ID of the i'th edge between them.

Network encoders: Let $\mathbf{Out}(v) = \{e_1, e_2, ..., e_d\}$ be the outgoing edges of node v. For an incoming edge e of v, v computes $\mathbf{b}(e) = T''(\mathbf{Out}(v), d)^{-1}\mathbf{t}''(e, d)$. For the coding coefficient $\beta(e, v, e_i)$ from e via v to e_i, v sets $\beta(e, v, e_i)$ to be the ith component of $\mathbf{b}(e)$.

Receiver decoder: The receiver receives

$$Y = TX, \tag{8}$$

where $T \in \mathbb{F}_q^{C\times C}$ can be indicated by the header of Y. If T is invertible the receiver can decode X correctly.

Thus, similar to RLNC [5], NRSC can be implemented in a distributed manner given that each node knows its local topology, *i.e.*, the adjacent neighbors. If an edge/node has been added/deleted, only local adjustments are needed.

E. Optimal throughput for multicast scenario

The theorem below shows that with high probability NRSC achieves the multicast capacity.

Theorem 18: With probability at least $1-C|\mathcal{E}|^4/q$, receiver r can decode X correctly.

Proof: Let *i.e.*, $\mathcal{X} = \{id(u, v), (u, v) \in \mathcal{V} \otimes \mathcal{V}\}$. Thus, $\mathcal{X}$ is the set of all random variables involved. By default we assume that any polynomial mentioned in the proof has variables in $\mathcal{X}$.

Let $det_G = \Pi_{u\in\mathcal{V}} det(u)$, where $det(u)$ is the determinant of the matrix $T''(\mathbf{Out}(u), |\mathbf{Out}(u)|)$ for node $u \in \mathcal{V}$. For each $u \in \mathcal{V}$, since each component of $T''(\mathbf{Out}(u), |\mathbf{Out}(u)|)$ is a polynomial of degree at most $|\mathbf{Out}(u)|$, $det(u)$ is a polynomial of degree at most $|\mathbf{Out}(u)|^2$. Thus det_G is a polynomial of degree at most $\sum_{u\in\mathcal{V}} |\mathbf{Out}(u)|^2 \le (\sum_{u\in\mathcal{V}} |\mathbf{Out}(u)|)^2 = |\mathcal{E}|^2$.

Let T be the transform matrix from s to r defined in Equation (8). We claim each element of $det_G \cdot T$ is a polynomial of degree at most $|\mathcal{E}|^4$. To see this, we first note that each component in $det(u) \cdot T''(\mathbf{Out}(u), |\mathbf{Out}(u)|)^{-1}$ is a polynomial of degree at most $|\mathbf{Out}(u)|^2 - |\mathbf{Out}(u)|$ (see Cramer's rule in [36]). Thus in the construction of NRSC, each local coding coefficient $\beta(e, u, e')$ used by $u \in \mathcal{V}$ is $Poly_{(e,u,e')}/det(u)$, where $Poly_{(e,u,e')}$ is a polynomial of degree at most $|\mathbf{Out}(u)|^2$. Each element in T can be expressed as $\sum_\alpha \bar{\beta}(\alpha)$, where $\bar{\beta}(\alpha) = \Pi_{(e,u,e')\in\alpha}\beta(e, u, e')$ and α is a path from s to r (see [5] for references). Thus each element in T can be expressed as $Poly_\alpha/(\Pi_{u\in\alpha} det(u))$, where $Poly_\alpha = \Pi_{(e,u,e')\in\alpha} Poly_{(e,u,e')}$. Thus $Poly_\alpha$ is a polynomial of degree at most $\sum_{u\in\alpha} |\mathbf{Out}(u)|^2 \le \sum_{u\in\mathcal{V}} |\mathbf{Out}(u)|^2 \le |\mathcal{E}|^2$. Since no node appears twice in a path of an acyclic network, det_G is divisible by $\Pi_{u\in\alpha} det(u)$ for each path α. Thus $det_G \sum_\alpha Poly_\alpha(\mathcal{X})/(\Pi_{u\in\alpha} det(u))$ is a polynomial of degree at most $|\mathcal{E}|^4$. This completes the proof of the claim that each element of $det_G \cdot T$ is a polynomial of degree at most $|\mathcal{E}|^4$.

Now we prove $det_G \cdot T$ is invertible with high probability. The determinant of $det_G \cdot T$ is denoted as det_r, which is therefore a polynomial of degree at most $|\mathcal{E}|^4 C$.

Without loss of generality let $\{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_C\}$ be the edge-disjoint paths from the source s to the receiver r. We first prove that det_r is a nonzero polynomial, *i.e.*, that there exists an evaluation of $\mathcal{X}$ such that $det_G \neq 0$ (*i.e.*, for each $u \in \mathcal{V}$ no two edges in $\mathbf{Out}(u)$ have the same ID) and the source can transmit C linearly independent packets via $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_C$.

We realize the evaluation of $\mathcal{X}$ as follows. First, assume each edge has a distinct ID. Second, since the ith outgoing edge of the source sends the ith row of $M = T''(\mathbf{Out}(s), C)^{-1}X$, the paths $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_C$ carry linearly independent packets on their initial edges. Third, the IDs of edges in $\mathcal{P}_i$ are all changed to be the ID of the first edge in $\mathcal{P}_i$. Note that this operation preserves the property that for each $u \in \mathcal{V}$ no two edges in $\mathbf{Out}(u)$ have the same ID (*i.e.*, $det_G \neq 0$). Finally, due to the code construction of NRSC (see Section VI-D), in fact the network uses routing to transmit the C independent source packets via $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_C$.

Thus under the evaluation of $\mathcal{X}$ above, the matrix $det_G \cdot T$ is invertible and therefore $det_r \neq 0$. Using Schwartz-Zippel Lemma [32], $det_r \neq 0$ and thus receiver r can decode X with probability at least $1-|\mathcal{E}|^4 C/q$ over all the possible evaluations of $\mathcal{X}$. □

Thus if the network has k receivers, using the union bound [33] on all receivers we conclude that with probability at least $1 - k|\mathcal{E}|^4 C/q$ each receiver can decode X.

Therefore the techniques over RLNC in multicast scenario can be directly moved into NRSC. For instance using network error-correcting codes [11], [12], NRSC is able to attain the optimal throughput for multicast with network errors.

F. IRVs under NRSC

Following Theorem 18 above, the relations between IRVs and the network structure can be shown to be the same as those for RLNC (see Lemma 1 for details). To be precise, for networks performing NRSC we have the following lemma.

Lemma 19:
1) The rank of the impulse response matrix $T'(\mathcal{Z})$ of an edge set $\mathcal{Z}$ with flow-rank z is at most z.
2) The IRVs of flow-independent edges are linearly independent with probability at least $1 - C|\mathcal{E}|^4/q$.

Proof: The proof is similar to the proof of Lemma 1. □

Note that for random error model (see Section II-E for details), all tomography schemes under RLNC are based on Lemma 1. Thus such schemes still work under NRSC.

VII. Locating adversarial errors under NRSC

In this section we show that the receivers in networks using NRSC are able to efficiently locate network adversaries even without prior knowledge of the network topology. The high level idea is that each column of error matrix plays the role of vector $\mathbf{e}$ for **RS-DECODE**$(H, \mathbf{e})$ (see Section II-I for details), where the columns of the Reed-Solomon parity-check matrix H comprise of the VIRVs of network edges. Thus the output of algorithm **RS-DECODE**$(H, \mathbf{e})$ locates the set of edges that introduce errors.

Assumptions and Justifications

1) At most z edges in $\mathcal{Z}$ suffer errors, *i.e.*, $\{e : e \in \mathcal{E}, \mathbf{z}(e) \neq 0\} = \mathcal{Z}$ and $|\mathcal{Z}| \le z$. When $2z + 1 \le C$, network error-correcting-codes (see Section II-G for details) are used so that the source message X is provably decodable.
2) Each node in $\mathcal{V} - \{r\}$ has out-degree at least $d = 2z$. Note that Theorem 13 proves it is a necessary condition for locating z errors.

Let the elements in $\mathcal{V} \otimes \mathcal{V}$ be indexed by $\{1, 2, ..., |\mathcal{V}|^2\}$. The parity check matrix $H \in \mathbb{F}_q^{d \times |\mathcal{V}|^2}$ is defined as $H = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{|\mathcal{V}|^2}]$, where $\mathbf{h}_i$ is the VIRV (with length d) of the ith element in $\mathcal{V} \otimes \mathcal{V}$. Then the adversarial error locating algorithm is constructed as follows.

Algorithm 6 LOCATE-ADV-RS

Input: Source matrix X, the parity-check matrix H, and the $C \times n$ matrix Y received by receiver r

$\mathcal{Z}' \leftarrow \emptyset$
Using network error correction code, decode X from Y
Compute $Y_{(RS,d)} = T''(\mathbf{In}(r), d)Y$
Compute $L = Y_{(RS,d)} - X_d$, where X_d comprises of the first d rows of X
for Each each column of L, say $\mathbf{v}$ **do**
  Compute $\mathbf{b} = \mathbf{RS\text{-}DECODE}(H, \mathbf{v})$
  if The ith component of $\mathbf{b}$ is nonzero **then**
    The ith node pair (u, v) in $\mathcal{V} \otimes \mathcal{V}$ is added as an edge $e = (u, v)$ into $\mathcal{Z}'$
  end if
end for
return $\mathcal{Z}'$

Theorem 20: The edge set $\mathcal{Z}'$ output by **LOCATE-ADV-RS** equals the actual error edge set $\mathcal{Z}$. The computational complexity of **LOCATE-ADV-RS** is $\mathcal{O}(n|\mathcal{V}|^2 d)$.

Before the proof we show the following key lemma when $|\mathbf{Out}(u)| \geq d$ for each node $u \in \mathcal{V} - \{r\}$. Recall that $\mathbf{z}(e)$ is the error packet injected on edge e.

Lemma 21: If the source message matrix X equals to 0,

$$Y_{(RS,d)} = \sum_{e \in \mathcal{E}} \mathbf{t}''(e, d)\mathbf{z}(e). \tag{9}$$

Proof: We proceed inductively. Throughout the proof let $\mathcal{E}_T$ be the set of edges satisfying the theorem, *i.e.*, $Y_{(RS,d)} = \sum_{e \in \mathcal{E}} \mathbf{t}''(e, d)\mathbf{z}(e)$ when $\mathbf{z}(e) = 0$ for all $e \in \mathcal{E} - \mathcal{E}_T$.

Step A: If $\mathcal{E}_T = \mathbf{In}(r)$, the theorem is true by the definition.

Step B: Since the network is acyclic, unless $\mathcal{E}_T = \mathcal{E}$, there must exist an edge $e \in \mathcal{E} - \mathcal{E}_T$ such that its adjacent outgoing edge set $\mathbf{Out}(e)$ is a subset of $\mathcal{E}_T$. Let $\mathbf{Out}(e) = \{e_1, e_2, ..., e_k\}$ with $k \geq d$. If only e suffers non-zero injected errors $\mathbf{z}(e)$, the output of e is $\mathbf{z}(e)$. Thus for each $i \in [1, k]$ the output of e_i is $\beta_i \mathbf{z}(e)$, where β_i is the ith component of $\mathbf{b}(e) = T''(\mathbf{Out}(e), k)^{-1}\mathbf{t}''(e, k)$ (see Section VI-D for details). Since $d \leq k$, we have $\sum_{i \in [1,k]} \beta_i \mathbf{t}''(e_i, d) = \mathbf{t}''(e, d)$. Since $\mathbf{Out}(e) \subseteq \mathcal{E}_T$, $Y_{(RS,d)} = \sum_{i \in [1,k]} \mathbf{t}''(e_i, d)\beta_i \mathbf{z}(e) = \mathbf{t}''(e, d)\mathbf{z}(e)$. Therefore Equation (9) is true for the case where only e suffers non-zero injected error $\mathbf{z}(e)$. Since NRSC is involves linear network coding, e can be added into $\mathcal{E}_T$.

Step C: Since the network is acyclic and each node (or edge) in $\mathcal{V}$ (or $\mathcal{E}$) is connected to r, we can repeat Step B until $\mathcal{E}_T = \mathcal{E}$. □

Recall the definition of IRV in Section III-A, we have $Y = \sum_{e \in \mathcal{E}} \mathbf{t}'(e)\mathbf{z}(e)$. Thus the following corollary is true for networks satisfying $|\mathbf{Out}(u)| \geq d$ for each node $u \in \mathcal{V} - \{r\}$:

Corollary 22: For each edge $e \in \mathcal{E}$, $T''(\mathbf{In}(r), d)\mathbf{t}'(e) = \mathbf{t}''(e, d)$.

Note that $C \geq d$. Thus, for the case where no error happens in the network and the source s transmits the $C \times n$ message matrix X, by Lemma 21 above we have $Y_{(RS,d)} = \sum_{i \in [1,C]} \mathbf{t}''(e_i, d)\mathbf{x}(e_i)$, where e_i is the ith edge of $\mathbf{Out}(s)$ and $\mathbf{x}(e_i)$ is the ith row of $M = T''(\mathbf{Out}(s), C)^{-1}X$ (see Section VI-D for details). Thus $Y_{(RS,d)} = T''(\mathbf{Out}(s), d)M = X_d$, where X_d is the matrix consisting of the first d rows of X.

Since NRSC involves linear network coding, we have the following corollary.

Corollary 23: When the source message is X, $Y_{(RS,d)} = X_d + \sum_{e \in \mathcal{E}} \mathbf{t}''(e, d)\mathbf{z}(e)$.

Then we can prove main theorem of this section as follows.

Proof of Theorem 20: Using Corollary 23 we have $L = \sum_{e \in \mathcal{Z}} \mathbf{t}''(e, d)\mathbf{z}(e)$. Since $|\mathcal{Z}| = z \leq d/2$, each column of L is a linear combination of at most $d/2$ columns of H. Additionally, since H is also a parity check matrix of a Reed-Solomon code, **RS-DECODE** correctly finds all the edges with nonzero injected errors, and therefore $\mathcal{Z}' = \mathcal{Z}$. For each column of L, **RS-DECODE** runs in time $\mathcal{O}(|\mathcal{V}|^2 d)$. Thus the overall time complexity of the algorithm is $\mathcal{O}(n|\mathcal{V}|^2 d)$. □

VIII. Topology estimation for networks with random errors under NRSC

Under NRSC, the section provides a lightweight topology estimation algorithm for the random error model. The high level idea is that once a candidate IRV is collected using **Algorithm 2 FIND-IRV** of Section IV-C, the corresponding VIRV can be computed by Corollary 22. Using the VIRV information, the corresponding edge can be detected. Thus **Algorithm 3 FIND-TOPO** is not involved.

For estimating the entire network topology, all assumptions in Section IV-C are required here except for Assumption 6, which assumes weak type common randomness.

Algorithm 7 FIND-TOPO-RS

Input: Matrixes $\{Y(i), i \in [1, t]\}$, which are the received matrix for source generations $\{1, 2, ..., t\}$.

$\mathcal{E}' \leftarrow \emptyset$
for $i = 1$ to t **do**
  Using network error-correction-code, compute the source message $X(i)$ in the i'th source generation
  Compute $E(i)_r = Y(i)_m - Y(i)_h X_m(i)$
end for
for $i, j = 1$ to t, and $i \neq j$ **do**
  if $\text{rank}(\mathbf{E(i)_r} \cap \mathbf{E(j)_r}) = 1$ **then**
    Let $\mathbf{h}$ be a vector in $\mathbf{E(i)_r} \cap \mathbf{E(j)_r}$
    Compute h_1 (and h_2) as the first (and second) component of $T''(\mathbf{In}(r), 2) \cdot \mathbf{h}$
    for $u, v \in \mathcal{V}$, and $u \neq v$ **do**
      If the ratio h_2/h_1 equals $id(u, v)$, add (u, v) as an edge into $\mathcal{E}'$
    end for
  end if
end for
return $\mathcal{E}'$

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the actual network topology, p_c be the probability defined in Assumption 4 of Section IV-C, p_s be $1-(1-z/q)[1-2C^2/(n-C)]$ and p'_a be $p_c+2p_s+C|\mathcal{E}|^4/q$. Then the theorem followed proves the correctness of **Algorithm 7**.

Theorem 24: 1) With probability at most $|\mathcal{V}|^2t^2/q$, $\mathcal{E}'$ has an edge which is not in $\mathcal{E}$.

2) If edge $e \in \mathcal{Z}(i) \cap \mathcal{Z}(j)$ for some $i \neq j \in \{1, 2, \ldots, t\}$, $e \in \mathcal{E}'$ with probability at least $1 - p'_a$.

Proof:

1) Consider node pair $(u, v) \in \mathcal{V} \otimes \mathcal{V}$ which is not in $\mathcal{E}$. Since $id(u, v)$ is independent from the network coding coefficients used in $\mathcal{G}$ and the random errors in each source generation, for line 11, the ratio h_2/h_1 is independent from $id(u, v)$. Thus $h_2/h_1 = id(u, v)$ with a probability at most $1/q$. Since there are at most t^2 pairs of i, j in $\{1, 2, \ldots, t\}$, using the union bound [33] edge $e(u, v)$ is accepted in $\mathcal{E}'$ with probability at most t^2/q. Since there are at most $|\mathcal{V}|^2$ node pairs, also by the union bound [33], $\mathcal{E}'$ has an edge which is not in $\mathcal{E}$ with probability at most $|\mathcal{V}|^2t^2/q$.
2) If $e \in \mathcal{Z}(i) \cap \mathcal{Z}(j)$, from the proof of Theorem 8 the intersection of $\mathbf{E(i)_r} \cap \mathbf{E(j)_r}$ equals $< \mathbf{t}'(e) >$ with probability at least $1 - p'_a$. Note that the difference between p_a in Theorem 8 and p'_a here comes from the difference between Lemma 1 (which is for RLNC) and Lemma 19 (which is for NRSC). Since each internal node has out-degree at least 2, from Corollary 22 we have $T''(\mathbf{In}(r), 2)\mathbf{t}'(e) = \mathbf{t}''(e, 2) = [id(e), (id(e))^2]^T$. Thus, in line 11 edge $e(u, v)$ would be accepted as a new edge in $\mathcal{E}'$. This completes the proof.

□

If the purpose of tomography is only to estimate the part of the topology that fails (*i.e.*, recovering the edges with errors), even Assumption 5 of Section IV-C is not needed anymore. Thus, **FIND-TOPO-RS** does not require that each edge suffers random errors with a non-negligible probability. Simply those edges suffering random errors can be directly detected via FIND-TOPO-RS, with high probability. For edges suffering random errors, **FIND-TOPO-RS** can detect them with high probability.

For the scenario where network edges (or nodes) suffer dynamic updates, **FIND-TOPO-RS** is more robust than the topology estimation algorithm under RLNC (see Section IV-C for details). The reason is that under RLNC the receiver must use algorithm **FIND-IRV** to recover all IRV information before proceeding the topology estimation algorithm **FIND-TOPO**. To be precise, under RLNC, it requires that the network keeps unchanged for $t = \Theta(\log(|\mathcal{E}|)|\mathcal{E}|)$ source generations (see the discussion after Theorem 8 for details). However, under NRSC, detecting an edge only requires that the network remains unchanged until such edges suffers two packet errors.

IX. Conclusion

This work examines passive network tomography on networks performing linear network coding in the presence of network errors. We consider both random and adversarial errors. In part I, under random linear network coding (RLNC) we give characterizations of when it is possible to find the topology, and thence the locations of the network errors. Under RLNC, many of the algorithms we provide have polynomial time computational complexity in the network size; for those that are not efficient, we prove intractability by showing reductions to computationally hard problems. In part II, we design network Reed-Solomon coding (NRSC) to address the undesirable tomography capabilities of RLNC under some (especially adversarial error) settings, while still preserving the key advantages of RLNC.

X. Acknowledgement

All the authors wish to thank the reviewers for their careful and diligent reading of this paper - their comments have significantly improved the results and the structure of this work. The authors also thank Alon Rosen for his comment about pesudorandoness, and Tracey Ho for her suggestions on approximate adversary localization under NRSC.

References

[1] H. Yao, S. Jaggi, and M. Chen, "Network coding tomography for network failures," in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, 2010.

[2] ——, "Network reed-solomon codes: Efficient byzantine adversary localization," in *Proc. of 44th Annual Asilomar Conference on Signals, Systems, and Computers, invited paper*, 2010.

[3] R. Castro, M. Coates, G. Liang, R. D. Nowak, and B. Yu, "Network tomography: recent developments," *Statistical Science*, 2004.

[4] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[5] T. Ho, R. Kötter, M. Médard, D. R. Karger, and M. Effros, "The Benefits of Coding over Routing in a Randomized Setting," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*, 2003.

[6] A.Le, and A.Markopoulou, "Locating Byzantine Attackers using SpaceMac", in *Proc. of IEEE Symposium on Network Coding (NetCod)*, 2010.

[7] T. Ho, M. Médard, R. Kötter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[8] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. of Allerton Conf. on Communications, Control, and Computing*, 2003.

[9] T. Ho, B. Leong, Y. H. Chang, Y. G. Wen, and R. Kötter, "Network monitoring in multicast networks using network coding," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*, 2005.

[10] G. Sharma, S. Jaggi, and B. K. Dey, "Network tomography via network coding," in *Proc. of Information Theory and Applications Workshop*, 2008.

[11] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Médard, "Resilient network coding in the presence of byzantine adversaries," in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, 2007.

[12] D. Silva, F. R. Kschischang, and R. Kötter, "A rank-metric approach to error control in random network coding," *IEEE Transactions on Information Theory*, vol. 54, no. 9, pp. 3951–3967, 2008.

[13] L. A. Adami, and B. A. Huberman, "Zipf's law and the Internet," *Glottometrics*, vol. 3, no. 9, pp. 143–150, 2002.

[14] C. Fragouli and A. Markopoulou, "A network coding approach to network monitoring," in *Proc. of Allerton Conf. on Communications, Control, and Computing*, 2005.

[15] C. Fragouli, A. Markopoulou, and S. Diggavi, "Topology inference using network coding," in *Proc. of Allerton Conf. on Communications, Control, and Computing*, 2006.

[16] M. Gjoka, C. Fragouli, P. Sattari, and A. Markopoulou, "Loss tomography in general topologies with network coding," in *Proc. of IEEE Global Communications Conference (Globecom)*, 2005.

[17] P. Sattari, A. Markopoulou, and C. Fragouli, "Multiple source multiple destination topology inference using network coding," in *Proc . of IEEE Symposium of Network Coding (NetCod)*, 2009.

[18] M. J. Siavoshani, C. Fragouli, S. Diggavi, and C. Gkantsidis, "Bottleneck discovery and overlay management in network coded peer-to-peer system," in *Proc. of ACM SIGCOMM workshop on Internet Network Management*, 2007.

[19] J. M. Siavoshani, C. Fragouli, and S. Diggavi, "Subspace properties of randomized network coding," in *Proc. of IEEE Information Theory Workshop (ITW)*, 2007.

[20] M. J. Siavoshani, C. Fragouli, and S. Diggavi, "On locating byzantine attackers," in *Network Coding Workshop: Theory and Applications*, 2008.

[21] Q. Wang, L. Vu, K. Nahrstedt, and H. Khurana, "Identifying malicious nodes in network-coding-based peer-to-peer streaming networks," in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, 2010.

[22] R. Diestel, *Graph Theory*. Springer-Verlag, Heidelberg, 2005.

[23] S-Y. R. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[24] D. Silva, F. R. Kschischang, and R. Kötter, "Capacity of random network coding under a probabilistic error model,," in *24th Biennial Symposium on Communications, Kingston, ON, Canada*, 2008.

[25] S. Katti, D. Katabi, H. Balakrishnan, and M. Medard, "Symbol-level network coding for wireless mesh networks," in *Proc. of ACM SIGCOMM*, 2008.

[26] S. Gollakota and D. Katabi, "Zigzag decoding: Combating hidden terminals in wireless networks," in *Proc. of ACM SIGCOMM*, 2008.

[27] I. Dumer, D. Micciancio, and M. Sudan, "Hardness of approximating the minimum distance of a linear code," *IEEE Transanction on Information Theory*, vol. 49, no. 1, pp. 22–37, 2003.

[28] A. Vardy, "The intractability of computing the minimum distance of a code," *IEEE Transanction on Information Theory*, vol. 43, no. 6, pp. 1757–1766, 1997.

[29] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM Journal of Applied Math*, vol. 8, pp. 300–304, 1960.

[30] U. K. Sorger, "A new reed-solomon code decoding algorithm based on newton's interpolation," *IEEE Transactions on Information Theory*, vol. 39, no. 2, pp. 358–365, 1993.

[31] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[32] J. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," *Journal of the ACM*, pp. 701–717, 1980.

[33] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[34] Y. Lindell and J. Katz, *Introduction to Modern Cryptography*. Chapman and Hall/CRC press, 2007.

[35] B. Thomas and V. Serge, "Proving the security of aes substitution-permutation network," in *Selected Areas in Cryptography, 12th International Workshop*, 2005.

[36] C. B. Boyer, *A History of Mathematics*, 2nd ed. Wiley, 1968.

[37] R. W. Yeung, *Information Theory and Network Coding*. Springer, 2008.

XI. Appendix

A. Network erasure model

An erasure on edge e means that the packet $\mathbf{x}(e)$ carried by e is treated as an all-zero length-n vector over $\mathbb{F}_q$ by the node receiving $\mathbf{x}(e)$, *i.e.*, the injected erroneous packet $\mathbf{z}(e)$ equals $-\mathbf{x}(e)$. Two network erasure models are considered:

1) *Random erasures*: Every edge e in $\mathcal{E}$ experiences erasures randomly and independently.

2) *Adversarial erasures*: The edges that suffer erasures are adversarially chosen.

B. Locating erasures under RLNC

The algorithm **LOCATE-RANDOM-RLNC** can be also generalized for locating network erasures (both random and adversarial), resulting in polynomial-time algorithms. Note that in the random error model the injected errors in Z are chosen at random, while in the random erasure model the injected errors are exactly the negative of the messages transferred. Thus Lemma 9 for random error model is not always true for the random erasure model. Hence, we need Lemma 25 below as an alternative.

Let $\mathcal{Z}$ be the set of edges suffering erasures and $|\mathcal{Z}| = z$. Let $\mathbf{t}(\mathbf{e}) \in \mathbb{F}_q^{1\times C}$ be the global encoding vector [37] of edge e, *i.e.*, the packet carried by e is $\mathbf{t}(\mathbf{e})X$ when no errors or erasures happen in the network. Let $T(\mathcal{Z}) \in \mathbb{F}_q^{z\times C}$ be the matrix whose rows comprise of $\{\mathbf{t}(\mathbf{e}), e \in \mathcal{Z}\}$. Recall that $E = T'(\mathcal{Z})Z$ (as defined in Equation (5)), where the rows of Z comprise of $\{\mathbf{z}(e) : e \in \mathcal{Z}\}$, *i.e.*, $\{-\mathbf{x}(e) : e \in \mathcal{Z}\}$. Then we have:

Lemma 25: If the source has max-flow z to the tails of the edges in $\mathcal{Z}$, with probability at least $1 - |\mathcal{E}|/q$, the matrix Z of injected errors has full row rank z and thus $\mathbf{E} = \mathbf{T}'(\mathcal{Z})$.

Proof: Since the network is directed and acyclic, for ease of analysis, we impose a partial order on the edges of $\mathcal{Z} = \{e_1, e_2, ..., e_z\}$. In particular, for any $i > j$, e_i cannot be upstream of e_j.

Similarly to Lemma 1, if the source has max-flow z to the tails of the edges in $\mathcal{Z}$, $T(\mathcal{Z})$ has full row rank z with a probability at least $1 - |\mathcal{E}|/q$ under RLNC.

The error corresponding to the erasure on e_1 equals $-\mathbf{t}(\mathbf{e_1})X$. The packet traversing e_2 may be effected by the first erasure. Hence, the error corresponding to the erasure on e_2 equals $-(\mathbf{t}(\mathbf{e_2}) - a_{1,2}\mathbf{t}(\mathbf{e_1}))X = -\bar{\mathbf{t}}(e_2)X$, where $a_{1,2} = c_{1,2}$ is the *unit effect from* e_1 *to* e_2. In general, the error corresponding to the erasure on e_i equals

$$\begin{aligned} -\bar{\mathbf{t}}(e_i)X &= -(\mathbf{t}(\mathbf{e_i}) - \sum_{j=1,2,...,i-1} c_{j,i}\bar{\mathbf{t}}(e_j))X \\ &= -(\mathbf{t}(\mathbf{e_i}) - \sum_{j=1,2,...,i-1} a_{j,i}\mathbf{t}(\mathbf{e_j}))X, \end{aligned}$$

where $c_{j,i}$ is the unit effect from e_j to e_i.

Thus $Z = -AT(\mathcal{Z})X$, where $A \in \mathbb{F}_q^{z\times z}$ and the (i,j)'th element of A equal $-a_{j,i}$ with $j < i$, 0 if $j > i$, 1 if $i = j$. Then A is invertible. If $T(\mathcal{Z})$ has full row rank z and X has an invertible $C \times C$ sub-matrix (for instance, the header corresponding to the identity matrix used in RLNC), Z has full row rank z. Thus, we have that $\mathbf{E} = \mathbf{T}'(\mathcal{Z})$. □

To locate *random erasures*, Lemma 25 proves that when the source has max-flow $|\mathcal{Z}|$ to the headers of $\mathcal{Z}$ who suffer erasures, $rank(Z) = z$ and $\mathbf{E} = \mathbf{T}'(\mathcal{Z})$. Thus **LOCATE-RANDOM-RLNC** can be used to locate erasures in the network, by using E in line 5 instead of E_r.

To use the efficient algorithm **LOCATE-RANDOM-RLNC** to locate *adversarial erasures*, by Lemma 25 it is required that every node has in-degree at least z. Otherwise, the high complexity algorithm **LOCATE-ADVERSARY-RLNC** can be used to find the locations of the adversarial erasures.

Finally, we note that the algorithm for locating erasures can also be used for locating edges experiencing problematic delays. Let $Y_d \in \mathbb{F}_q^{C\times n}$ be the delayed packet matrix received by r. Then r can locate the delayed edges by treating Y_d as the erasure matrix E and then using the scheme for locating network erasures.

Hongyi Yao (M'10) received a Ph.D. (2010) and B.S. (2007) from Tsinghua University, Beijing, China, in 2007 and 2010.

He is currently a postdoc with Professor Tracey Ho at California Institute of Technology.His research interests include secure network transmission, reliable network control and secure wireless communications.

Sidharth Jaggi (M'00) is an Assistant Professor in Information Engineering at the Chinese University of Hong Kong. He received a Ph.D. (2006) and M.S. (2001) from the California Institute of Technology, and a B.Tech. degree (2000) in Electrical Engineering from the Indian Institute of Technology, Bombay. His primary research interests are in network coding, coding theory and information theory.

Minghua Chen (S'04) received his B.Eng. and M.S. degrees from the Department of Electronics Engineering at Tsinghua University in 1999 and 2001, respectively. He received his Ph.D. degree from the Department of Electrical Engineering and Computer Sciences at University of California at Berkeley in 2006. He spent one year visiting Microsoft Research Redmond as a Postdoc Researcher. He joined the Department of Information Engineering, the Chinese University of Hong Kong, in 2007, where he currently is an Assistant Professor. He received the Eli Jury award from UC Berkeley in 2007 (presented to a graduate student or recent alumnus for outstanding achievement in the area of Systems, Communications, Control, or Signal Processing), the ICME Best Paper Award in 2009, and the IEEE Transactions on Multimedia Prize Paper Award in 2009. His research interests include resource provisioning for data centers and power systems, distributed and stochastic network optimization and control, multimedia network