Network Monitoring in Multicast Networks Using Network Coding

Tracey Ho
Coordinated Science Laboratory
University of Illinois
Urbana, IL 61801
Email: trace@mit.edu

Ben Leong, Yu-Han Chang, Yonggang Wen
CSAIL & LIDS
Massachusetts Institute of Technology
Cambridge, MA 02139
Email: {benleong, ychang, eewyg}@mit.edu

Ralf Koetter
Coordinated Science Laboratory
University of Illinois
Urbana, IL 61801
Email: koetter@csl.uiuc.edu

***Abstract*— In this paper we show how information contained in robust network codes can be used for passive inference of possible locations of link failures or losses in a network. For distributed randomized network coding, we bound the probability of being able to distinguish among a given set of failure events, and give some experimental results for one and two link failures in randomly generated networks. We also bound the required field size and complexity for designing a robust network code that distinguishes among a given set of failure events.**

I. Introduction

The distributed randomized network coding approach of [1] provides a simple way to achieve robustness to link failures in multisource multicast. In this approach, interior network nodes independently and randomly choose linear mappings from inputs to outputs. Coefficient vectors specifying the aggregate linear combinations are sent with each data block or packet, allowing decoding at the sinks under different combinations of link failures or packet losses, as long as the remaining rate is sufficient.

In this paper we make the observation that the coefficient vectors transmitted in a distributed randomized network coding setup are not simply a necessary overhead for recovery of the coded messages at the sinks, but may be used to deduce, additionally, useful information on the location of link failures or packet losses. This is because losses on different links affect the coefficient vectors obtained at the sinks differently. Knowledge of the original network topology and network code allows inference, from changes in the coefficient vectors obtained at the sinks, of possible locations of losses in the network.

The problem of monitoring interior network state or performance parameters such as link failures, loss rates, or delays using end-to-end observations is commonly known as network tomography, based on the analogy with the medical tomography problem of non-intrusive imaging. Such monitoring can be useful for network maintenance or management. In our case, we have a form of passive network tomography, since our inferences are based on passive end-to-end observations of existing network traffic, rather than the use of active probes. In particular, the network code coefficient vectors play double duty by allowing link failure monitoring in addition to allowing the sink nodes to correctly decode the incoming data under different failure patterns.

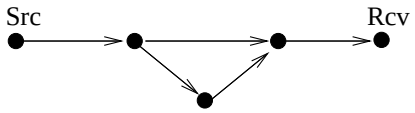

Fig. 1. An example in which distinct logical link segments are indistinguishable with only end-to-end measurements. Here failure of either the left-most or right-most link results in the same observation at the sink.

In this paper, we look at the extent to which network codes allow us to distinguish among different *failure patterns*, i.e. sets of links that fail simultaneously. Failure patterns that result in different sink observations under a network code can be distinguished from each other by that code. As in all end-to-end tomography approaches, failures can be localized only up to segments of the *logical topology*, which are segments between branches in the network. Furthermore, for a given topology and given source and sink locations, failures of distinct logical segments may be indistinguishable by end-to-end observations if the segments lie on exactly the same set of source-sink paths, since the corresponding end-to-end observations are identical. This is illustrated in Figure 1. For a given network and given locations of sources and sinks, we consider two failure patterns p_1 and p_2 to be *indistinguishable* if the set of source-sink paths containing at least one link in p_1 is identical to the set of source-sink paths containing at least one link in p_2, and *distinguishable* otherwise.

The problem we consider here is in some sense the opposite of that in [2], which also deals with link failures, but is concerned with the minimum number of different groups of failure patterns the network needs to be aware of, representing the minimum amount of control information needed to ensure continued transmission of data across different failure patterns. This paper, on the other hand, is concerned with how many failures patterns can be distinguished.

The rest of the paper is organized as follows: Section II provides some background and a brief overview of related work, Section III describes our network model, Section IV gives the main results, Section V gives our mathematical development and proofs, Section VI gives some simulation results, and Section VII concludes the paper with a summary of the results and a discussion of further work.

II. Background and related work

A. Network coding

The field of network coding has its origins in the work of Ahlswede et al. [3] and Li et al. [4]. Li et al. [4] prove that linear coding with finite symbol size is sufficient for multicast connections, showing that network codes may potentially be simple and practical. Koetter and Médard [5] present an algebraic framework for linear network coding, which provides the basic model and mathematical foundation for this work. They also demonstrate that network coding can be used to provide robust solutions to multi-source multicast networks with link failures, in which only sink nodes need to change behavior in response to different failures.

Distributed randomized network coding, introduced in Ho et al. [1], gives an approach for robust multi-source multicast in a distributed setting. In this technique, nodes independently select random linear mappings from inputs onto outputs over some finite field, which achieves all feasible connections with probability tending to 1 as the field size grows. The sinks need only know the overall linear combination of source processes in each of their incoming signals. This information is sent through the network as a vector, for each signal, of coefficients corresponding to each of the source processes, updated at each coding node by applying the same linear mappings to the coefficient vectors as to the information signals. Robustness to link failures and errors in the random selection of codes improves with excess capacity in the network [6]. Chou et al. [7] have proposed and demonstrated by simulation a packet-based implementation in which source packets are divided into generations, and only packets in the same generation are linearly combined.

Concurrent independent work by Sanders et al. [8] and Jaggi et al. [9] considers multicast on acyclic delay-free graphs, giving centralized deterministic and randomized polynomial-time algorithms for finding network coding solutions over a subgraph consisting of flow solutions to each sink.

B. Network tomography

The problem of monitoring or inference of internal network characteristics from end to end measurements is commonly known as network tomography. Such characteristics include link status, losses and delay characteristics.

This problem has been considered by a large number of papers, some of which are surveyed in [10]. Existing approaches may be classified in various ways. Active measurement approaches, such as those of [11], [12], involve sending additional probe traffic, whereas passive approaches [13] aim to infer network characteristics from existing network traffic. Approaches may also be classified into unicast [14], [12] or multicast [15] approaches, depending on whether unicast or multicast traffic/probes are used. While many existing works focus on wired networks, network tomography for sensor networks has also been considered [16].

III. Model

We consider networks with directed error-free links. Network coding is done only at branch nodes (i.e. nodes with degree three or more), as there is no capacity or reliability advantage from coding within a logical link. Thus, in the rest of the paper we will consider logical links rather than physical links, using the terms 'link' and 'logical link' interchangeably.

Our basic mathematical framework and model is based on that in [3], [5]. A network is represented as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of network nodes and $\mathcal{E}$ is the set of links, such that information can be sent noiselessly from node i to j for all $(i, j) \in \mathcal{E}$, and $|\mathcal{E}| = \eta$. i and j are called the *origin* and *destination* respectively of link (i, j). The origin and destination of a link $l \in \mathcal{E}$ are denoted $o(l)$ and $d(l)$ respectively.

Link l is an *incident outgoing link* of node v if $v = o(l)$, and an *incident incoming link* of v if $v = d(l)$. We call an incident incoming link of a sink node a *terminal link*. Link l carries the random process $Y(l)$.

There are r independent information sources $X_1, X_2, \ldots, X_r$. Source X_i is generated at a node $\alpha_i \in \mathcal{E}$, and multicast to all nodes $j \in \{\beta_1, \ldots, \beta_d\}$ for all $i \in [1, r]$, which we refer to as multi-source multicast. The nodes $\alpha_1, \ldots, \alpha_r$ are called *source nodes* and the d nodes $\beta_1, \ldots, \beta_d$ are called *sink nodes*.

A multicast transmission problem is specified by a network $\mathcal{G}$, source and sink locations, and source rates. For a given transmission problem, a network code is *valid* if every sink is able to reconstruct all the source information without error. Failure of a link l is modeled as removal of l from $\mathcal{G}$.

We assume that the time unit is chosen such that the capacity of each link is one bit per unit time, and the random processes X_i have a constant bit and entropy rate of one bit per unit time. Edges with larger capacities are modelled as parallel edges, and sources of larger entropy rate are modelled as multiple sources at the same node.

The processes X_i, $Y(l)$ generate binary sequences. We assume that information is transmitted as vectors of bits which are of equal length u, represented as elements in the finite field $\mathbb{F}_{2^u}$. The length of the vectors is equal in all transmissions, and all links are assumed to be synchronized with respect to the symbol timing.

For simplicity we consider linear coding[1] on acyclic delay-free graphs; burst [4] or pipelined [8] network codes on acyclic networks with link delays are essentially equivalent. For cyclic networks, such as the random geometric graphs in our simulations of Section VI, our approach is to code over an acyclic subgraph. This is in some cases suboptimal compared to cyclic coding approaches such as those of [5], [4], but the overhead of specifying such codes is higher, which is less attractive for non-static networks. In a linear code, the signal $Y(j)$ on a link j is a linear combination of processes X_i generated at node $v = o(j)$ and signals $Y(l)$ on links l such

[1] which is sufficient for multicast [4]

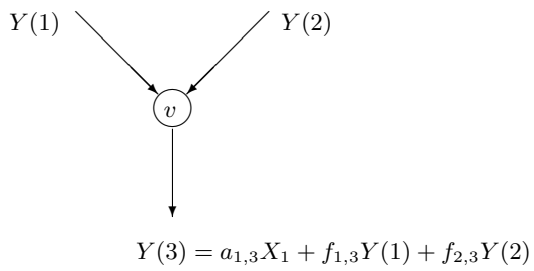

Fig. 2. Illustration of linear coding at a node.

that $d(l) = o(j)$:

$$Y(j) = \sum_{\{i\,:\,\alpha_i = v\}} a_{i,j} X_i + \sum_{\{l\,:\,d(l) = v\}} f_{l,j} Y(l)$$

IV. Main results

Suppose an acyclic graph, source and sink locations, and link failure statistics are given, and suppose we are interested in identifying some of the more likely failure events. For instance, if each failure occurs with relatively small probability, we may wish to distinguish among failure patterns that consist of up to some small number of links. Among these patterns of interest, there may be some that are indistinguishable from each other as noted earlier with the example of Figure 1. Sets of indistinguishable failure patterns are grouped together into failure events, and the remaining failure patterns are considered individual failure events. For a given set $\mathcal{C}$ of distinguishable failure events, and a given failure event $c \in \mathcal{C}$, a *monitoring ambiguity* is said to exist for a network code if the corresponding coefficient vectors on the terminal links are identical to those for some other failure event in $\mathcal{C}$. This definition is independent of the actual method of inferring failure events from sink observations; one basic approach is to have a lookup table of the observations corresponding to different failure/loss events.

Let L be the maximum number of logical links on a source-sink path, $\mathcal{S}$ the set of sources, and $\mathcal{T}$ the set of terminal links.

Theorem 1 gives an upper bound on the probability of a monitoring ambiguity for a given link in a given problem, which decreases approximately inversely with field size $q = 2^u$ or exponentially with code length u. The bound is very general, depending only on three parameters, $|\mathcal{C}|$, L and q, and is correspondingly pessimistic for networks that are not worst-case examples. In Section VI, we show by simulation much better performance on randomly generated geometric graphs.

Theorem 1: For a given set $\mathcal{C}$ of distinguishable failure events, and a given failure event $c \in \mathcal{C}$, the probability of a monitoring ambiguity in a random linear network code is at most $1 - \left(1 - \frac{|\mathcal{C}|-1}{q}\right)^L$. □

In the case where rerouting or further testing following a monitoring ambiguity is undesirable, we may wish to have a network code that is valid under all failure events in a set $\mathcal{C}$, and distinguishes among them without any monitoring ambiguities. A simple lower bound on the minimum field size q needed for such a code is

$$q \geq |\mathcal{C}|^{\frac{1}{rt}}$$

where r and t are the number of sources and terminal links respectively. This is obtained by noting that $|\mathcal{C}|$ cannot exceed the maximum number of possible values q^{rt} of the coefficient vectors of the terminal links. An upper bound is given by the following theorem.

Theorem 2: Suppose $\mathcal{C}$ is a set of distinguishable failure events. A network code that is valid under all failure events in $\mathcal{C}$ and distinguishes among them without any monitoring ambiguities can be obtained in any finite field of size greater than $|\mathcal{C}|(\frac{|\mathcal{C}|-1}{2} + d)$. □

Theorem 3: Suppose $\mathcal{C}$ is a set of distinguishable failure events. A valid network code of field size $\gamma|\mathcal{C}|(\frac{|\mathcal{C}|-1}{2} + d)$ network code that is valid under all failure events in $\mathcal{C}$ and distinguishes among them without any monitoring ambiguities can be obtained using a randomized procedure in expected time $O\left(\left(\frac{\gamma}{\gamma-1}\right)^\eta |\mathcal{C}| \left(\eta I r + d r^{2.376} + |\mathcal{C}| r t\right)\right)$, where I is the maximum in-degree of a node. □

Again, these are very general bounds; we would expect to do substantially better on many networks.

V. Mathematical development and proofs

Our proofs use the following lemma from [17], which, like the Schwartz-Zippel Lemma, bounds the probability that a multivariate polynomial evaluated at a random point equals zero, but is tighter as it takes into account the maximum degree of each variable. Its proof in [17] uses the Schwartz-Zippel Lemma.

Lemma 1: Let P be a nonzero polynomial in $\mathbb{F}[\xi_1, \xi_2, \ldots]$ in which the largest exponent of any variable ξ_i is at most x, and whose total degree is less than or equal to xy. Values for $\xi_1, \xi_2, \ldots$ are chosen independently and uniformly at random from $\mathbb{F}_q \subseteq \mathbb{F}$. The probability that P equals zero is at most $1 - (1 - x/q)^y$ for $x < q$. □

For each $c \in \mathcal{C}$, denote by $E_c(s, l)$ the difference in the coefficient corresponding to source s in the coefficient vector of terminal link l due to failure of links in c, in terms of code coefficients $\{a_{s,i}, f_{i,j}\}$.

Lemma 2: $E_c(s, l)$ has maximum degree L and is linear in each variable $\{a_{s,i}, f_{i,j}\}$.

Proof: Note that $E_c(s, l)$ is the sum of the contributions of all paths from s to l that pass through one or more links in c, where the contribution of a path from s through links $l_1, l_2, \ldots, l_x, l$ in order is the product $a_{s,l_1} f_{l_1,l_2} f_{l_2,l_3} \ldots f_{l_x,l}$. ■

Proof of Theorem 1: Consider a failure event $c \in \mathcal{C}$. The probability of monitoring ambiguity for c is given by

$$\Pr(\exists c' \in \mathcal{C}, c' \neq c, \text{s.t. } E_{c'}(s, l) = E_c(s, l)\ \forall\ s \in \mathcal{S}, l \in \mathcal{T}).$$

This is upper bounded by

$$\Pr\left(\exists c' \in \mathcal{C}, c' \neq c, \text{s.t. } E_{c'}(s_{c,c'}, l_{c,c'}) = E_c(s_{c,c'}, l_{c,c'})\right)$$
$$= \Pr\left(\prod_{c' \in \mathcal{C}:\ c' \neq c} \left(E_{c'}(s_{c,c'}, l_{c,c'}) - E_c(s_{c,c'}, l_{c,c'})\right) = 0\right),$$

where $s_{c,c'}, l_{c,c'}$ are respectively some source and terminal link for which $E_{c'}(s_{c,c'}, l_{c,c'})$ is not identically equal to $E_c(s_{c,c'}, l_{c,c'})$. Now, by Lemma 2, each difference term $(E_{c'}(s_{c,c'}, l_{c,c'}) - E_c(s_{c,c'}, l_{c,c'}))$ has maximum degree L and is linear in each variable $\{a_{s,i}, f_{i,j}\}$. The product of $|\mathcal{C}| - 1$ such terms thus has maximum degree $(|\mathcal{C}| - 1)L$, and the largest exponent of any variable is at most $|\mathcal{C}| - 1$. Therefore, by Lemma 1, the probability that their product equals zero is at most $1 - \left(1 - \frac{|\mathcal{C}|-1}{q}\right)^L$. ■

Proof of Theorem 2: Firstly, we want the network code to be valid for each error event $c \in \mathcal{C}$. This condition is equivalent to the product of the transfer matrix determinants of all d sinks for all $c \in \mathcal{C}$ being nonzero [5]. This product P_1 is an expression of total degree at most $|\mathcal{C}|d\eta$, in which the largest exponent of any variable is at most $|\mathcal{C}|d$ [1].

For the network code to be able to distinguish among all failure combinations in $\mathcal{C}$, we must have, for each pair of distinct $c, c' \in \mathcal{C}$,

$$E_c(s_{c,c'}, l_{c,c'}) - E_{c'}(s_{c,c'}, l_{c,c'}) \neq 0$$

for some source $s_{c,c'}$ and terminal link $l_{c,c'}$. By Lemma 2, each difference term $(E_c(s_{c,c'}, l_{c,c'}) - E_{c'}(s_{c,c'}, l_{c,c'}))$ has maximum degree L and is linear in each variable $\{a_{s,i}, f_{i,j}\}$. There are at most $\binom{|\mathcal{C}|}{2} = \frac{|\mathcal{C}|(|\mathcal{C}|-1)}{2}$ distinct difference terms whose product must be nonzero. This product P_2 has degree at most $\frac{|\mathcal{C}|(|\mathcal{C}|-1)L}{2}$, and the largest exponent of any variable is at most $\frac{|\mathcal{C}|(|\mathcal{C}|-1)}{2}$.

The largest exponent of any variable in the product P_1P_2 is $|\mathcal{C}|(\frac{|\mathcal{C}|-1}{2} + d)$. By Lemma 1, P_1P_2 has positive probability of being nonzero when values for variables $f_{i,j}$ are chosen uniformly at random from a field of size greater than $|\mathcal{C}|(\frac{|\mathcal{C}|-1}{2} + d)$. Thus, in any finite field of size larger than $|\mathcal{C}|(\frac{|\mathcal{C}|-1}{2} + d)$, there exists an assignment of values for variables $f_{i,j}$ such that $P_1P_2 \neq 0$. ■

Proof of Theorem 3: Consider a randomized procedure that simultaneously chooses values for variables $f_{i,j}$ uniformly at random from a field of size $q = \gamma|\mathcal{C}|(\frac{|\mathcal{C}|-1}{2} + d)$. For each such assignment of values, it checks whether the resulting network code is valid for all failure events in $\mathcal{C}$ and distinguishes among them without monitoring ambiguities; if the code does not satisfy these properties, the process is repeated with another randomly chosen set of values. One way to check if a network code is valid for a failure event is to compute the coefficient vectors of all links, which takes $O(\eta Ir)$ time, and to check whether the coefficient vectors of the terminal links of each sink have full rank, which takes $O(dr^{2.376})$ time. To check if the network code distinguishes between all failure combinations in $\mathcal{C}$, we can compute E_c for each $c \in \mathcal{C}$, which takes $O(|\mathcal{C}|\eta Ir)$ time in total, and check that they are all distinct, which takes $O(|\mathcal{C}|^2 rt)$ time.

By Lemma 1, each random assignment of values is successful with probability at least $\left(1 - \frac{1}{\gamma}\right)^\eta$. Since $\left(\frac{\gamma}{\gamma-1}\right)^\eta$ tries are required in expectation, the total expected execution time is $O\left(\left(\frac{\gamma}{\gamma-1}\right)^\eta |\mathcal{C}| \left(\eta Ir + dr^{2.376} + |\mathcal{C}|rt\right)\right)$. ■

VI. Simulations

Since our bounds are pessimistic except for worst-case networks, we have run simulations to give an idea of actual performance on random geometric graphs. The simulations do not attempt to characterize precisely the achievable monitoring performance, but seek to give an idea of the performance of random network coding with short code lengths.

Our experiments are run on 15-node random geometric networks with 2 sources and 2 sinks, generated by scattering nodes randomly over a unit square and connecting nodes within range ρ of each other. The parameter values for the tests are chosen such that the resulting random graphs are in general connected and able to support the desired connections, while being small enough for the simulations to run efficiently. For each network, we use a simple randomized algorithm to generate acyclic digraphs to disallow the transmission of information in cycles. Distributed randomized network coding is run over the resulting network; networks for which we cannot find a valid solution are discarded.

We repeated this process 1,000 times and generated 923 feasible 15-node random networks with 2 sources and 2 sinks. We ran 10 trials on each network. We consider two cases: first, taking the set of possible failure events $\mathcal{P}$ as the set of all failures of individual links; second, taking the set $\mathcal{P}$ as the set of all failures of one or two links. The raw results for these two cases on the random networks, using a finite field of size 61, are shown as scatterplots in Figures 3 and 4 respectively. Since we do not determine for each graph which failure events are distinguishable, $\mathcal{P}$ may contain failure events that are indistinguishable. Our results thus give a pessimistic estimate of the probability of distinguishing among distinguishable failure patterns of one or two links. The probability of unambiguity when $\mathcal{P}$ contains only single link failure events is above 90% in general, while the corresponding probability of unambiguity is somewhat lower when $\mathcal{P}$ contains failure events involving up to two links.

The aggregate fitted results for these networks with other finite field sizes for up to two link failures are shown in Figure 5. They show that the probability of unambiguity increases, with diminishing returns, as q increases. The apparent anomaly for networks with node degree higher than 9 is probably due to experimental error since our sample set contains few such networks. Repeating the experiments with $q = 7917$ and $q = 40,009$ yielded results almost indistinguishable from those for $q = 61$. For these networks, $|\mathcal{P}|$ ranges from about 300 (at node degree 3) to about 2,700 (at node degree 10). It is apparent from these results that even for $q \ll |\mathcal{P}|$, the probability of unambiguity is still relatively high.

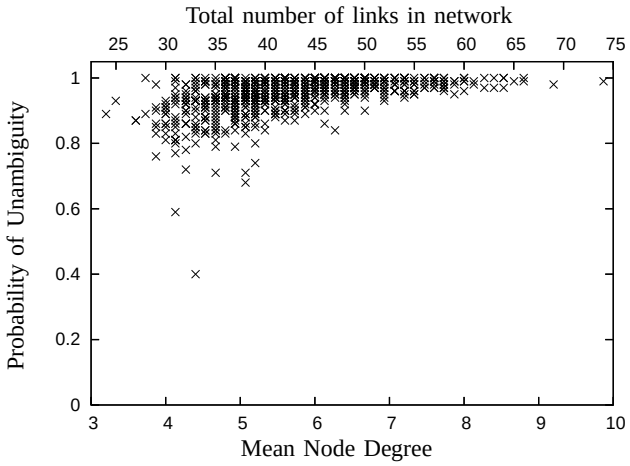

Fig. 3. Scatterplot for 923 15-node networks with 2 sources and 2 sinks for one link failure for $q = 61$.

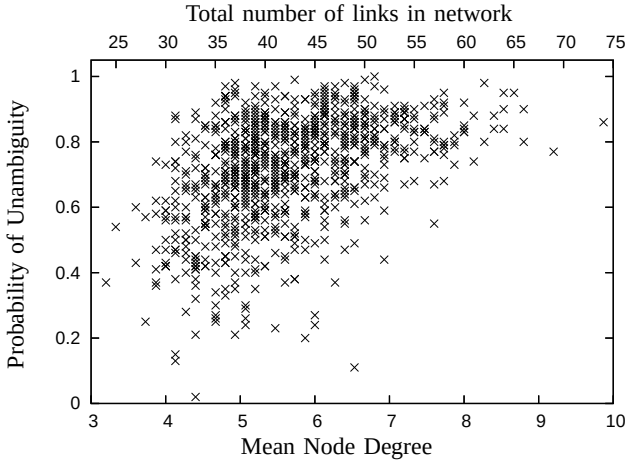

Fig. 4. Scatterplot for 923 15-node networks with 2 sources and 2 sinks for up to two link failures for $q = 61$.

An increase in the probability of unambiguity with mean node degree can also be observed. This is because there are fewer indistinguishable failure patterns of one to two links in more densely-connected networks.

Experiments with random networks with different numbers of sources r and number of sinks d exhibited a similar trend.

VII. Conclusions and further work

We have shown how information contained in robust distributed network codes can be used to deduce possible locations of link failures or losses in a network, without the overhead of additional probes. We provide worst-case bounds regarding the relationship between failure ambiguity and the coding field size, and we characterize this relationship in more benign networks with experimental simulations. We also bound the required field size and complexity for designing a robust network code that distinguishes among a given set of failure events.

Further work includes extensions to network coding in other network settings such as wireless and non-multicast. For networks not using network coding, we seek to explore the potential benefits of using network coding in active probing schemes.

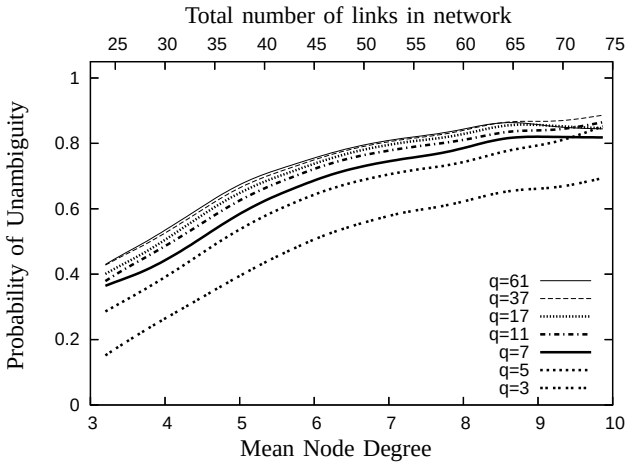

Fig. 5. Plot of mean probability of ambiguity against mean node degree for 923 15-node networks with 2 sources and 2 sinks for up to two link failures for different field sizes q.

References

[1] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proceedings of 2003 IEEE International Symposium on Information Theory*, June 2003.

[2] T. Ho, M. Medard, and R. Koetter, "An information-theoretic view of network management," *IEEE Transactions on Information Theory*, vol. 51, no. 4, 2005.

[3] 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.

[4] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, pp. 371–381, 2003.

[5] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, October 2003.

[6] T. Ho, M. Médard, J. Shi, M. Effros, and D. R. Karger, "On randomized network coding," in *Proceedings of 41st Annual Allerton Conference on Communication, Control, and Computing*, October 2003.

[7] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proceedings of 41st Annual Allerton Conference on Communication, Control, and Computing*, October 2003.

[8] P. Sanders, S. Egner, and L. Tolhuizen, "Polynomial time algorithms for network information flow," in *15th ACM Symposium on Parallel Algorithms and Architectures*, 2003, pp. 286–294.

[9] S. Jaggi, P.A. Chou, and K. Jain, "Low complexity algebraic network codes," in *Proceedings of the IEEE International Symposium on Information Theory*, June 2003.

[10] M. Coates, A. Hero, R. Nowak, and B. Yu, "Internet tomography," 2002.

[11] M. Coates, M. Rabbat, and R. Nowak, "Merging logical topologies using end-to-end measurements," in *IMC*, 2003.

[12] Y. Chen, D. Bindel, H. Song, and R. Katz, "An algebraic approach to practical and scalable overlay monitoring," in *Proceedings of SIGCOMM*, August 2004.

[13] Y. Tsang, M. Coates, and R. Nowak, "Passive unicast network tomography using EM algorithms," in *IEEE Int'l Conference on Acoustics, Speech, and Signal Processing*, May 2001.

[14] N. G. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley, "Network delay tomography from end-to-end unicast measurements," *Lecture Notes in Computer Science*, vol. 2170, 2001.

[15] N. Duffield, J. Horowitz, F. Presti, and D. Towsley, "Multicast topology inference from measured end-to-end loss," *IEEE Transactions on Information Theory*, vol. 48, no. 1, pp. 26–45, 2002.

[16] G. Hartl and B. Li, "Loss inference in wireless sensor networks," in *Proceedings of IPSN*, April 2004.

[17] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "Toward a random operation of networks," *IEEE Transactions on Information Theory, submitted*, 2003.