References

[1] S. Yang and R. W. Yeung, "Batched Sparse Codes," *Proceedings of the 2011 IEEE International Symposium on Information Theory Proceedings, ISIT*, pp. 2647–2651, 2011.

[2] R. Ahlswede, N. Cai, S. R. Li and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, 2000.

[3] C. Fragouli and E. Soljanin, "Network coding fundamentals," *Monograph in Series, Foundations and Trends in Networking*, vol. 2, pp. 1–133, 2007.

[4] T. Matsuda, T. Noguchi and T. Takine, "Survey of Network Coding and Its Applications," *IEICE Transactions*, pp. 698–717, 2011.

[5] T. Ho, R. Koetter, M. Medard, D. R. Karger and M. Effros, "The benefits of coding over routing in a randomized setting," *Proceedings of the IEEE International Symposium on Information Theory*, p. 442, 2003.

[6] R. Koetter and M. Medard, "Beyond routing: an algebraic approach to network coding," *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2002*, vol. 1, pp. 122–130, 2002.

[7] J. K. Sundararajan, D. Shah and M. Medard, "Online network coding for optimal throughput and delay – the two-receiver case," *Computing Research Repository (CoRR)*, vol. abs/0806.4264, 2008.

[8] P. A. Chou, Y. Wu and K. Jain, "Practical network coding," *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, vol. 41, pp. 40–49, 2003.

[9] H. Lin, Y. Lin and H. Kang, "Adaptive Network Coding for Broadband Wireless Access Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 1, pp. 4–18, 2012.

[10] H. Shojania and B. Li, "Parallelized Progressive Network Coding With Hardware Acceleration," *Proceedings of the 2007 IEEE International Workshop on Quality of Service*, pp. 47–55, 2007.

[11] P. Sadeghi, R. Shams and D. Traskov, "An Optimal Adaptive Network Coding Scheme for Minimizing Decoding Delay in Broadcast Erasure Channels," *EURASIP Journal of Wireless Communications and Networking*, *Special Issue on Network Coding for Wireless Communication* vol. 2010, pp. 1–14, 2010.

[12] P. Maymounkov and N. J. A., "Methods for Efficient Network Coding," *Proceedings of 44th Annual Allerton Conference on Communication, Control, and Computing*, vol. 1, pp. 482–491, 2006.

[13] J. Heide, M. V. Pedersen and F. H. P. Fitzek, "Decoding algorithms for random linear network codes," *Proceedings of the IFIP TC 6th international conference on Networking*, pp. 129–136, 2011.

[14] R. W. Yeung and N. Cai, "Network error correction, part I: Basic concepts and upper bounds," *Communications in Information and Systems*, vol. 6, pp. 19–36, 2006.

[15] N. Cai and R. W. Yeung, "Network error correction, part II: lower bounds," *Communications in Information and Systems*, vol. 6, pp. 37–54, 2006.

[16] R. W. Yeung, S. R. Li, N. Cai and Z. Zhang, "Network coding theory," *Foundation and Trends in Communications and Information Theory*, vol. 2, pp. 241–381, 2005.

[17] R. W. Yeung, *Information Theory and Network Coding*, 1st ed., Springer Publishing Company Incorporated, New York, 2008.

[18] B. Shrader and A. Ephremides, "On packet lengths and overhead for random linear coding over the erasure channel," *Computing Research Repository (CoRR)*, vol. abs/0704.0831, 2007.

[19] R. Koetter and F. R. Kschischang, "Coding for Errors and Erasures in Random Network Coding," *IEEE Transactions on Information Theory*, vol. 54, pp. 3579–3591, 2008.

[20] T. Ho, "Networking from a network coding perspective," Ph.D. dissertation, Dept. Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 2005.

[21] D. Silva and F. R. Kschischang, "Using Rank-Metric Codes for Error Correction in Random Network Coding," *Proceedings of the IEEE International Symposium on Information Theory, ISIT 2007*, pp. 796–800, 2007.

[22] D. Silva, F. R. Kschischang and R. Koetter, "A Rank-Metric Approach to Error Control in Random Network Coding," *Proceedings of the IEEE Information Theory Workshop on Information Theory for Wireless Networks*, pp. 1–5, 2007.

[23] N. Cai and R. W. Yeung, "Network Coding and Error Correction," *Proceedings of the IEEE Information Theory Workshop*, pp. 119–122, 2002.

[24] H. Wang, J. Liang and J. Kuo, "Overview of robust video streaming with network coding," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, pp. 36–50, 2010.

[25] D. J. C. Mackay, *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

[26] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*, University of Illinois Press, Cambridge, 1963.

[27] M. V. Pedersen, J. Heide, F. H. P. Fitzek and T. Larsen, "Network Coding for Mobile Devices - Systematic Binary Random Rateless Codes," *Proceedings of the IEEE International Conference on Communication (ICC) – ICC09*, pp. 1–6, 2009.

[28] S. Lin and D. J. Costello, *Error Control Coding, Second ed.,* Prentice-Hall, Inc.,New Jersey, 2004.

[29] O. Pretzel, *Error-correcting codes and finite fields (student ed.)*, Oxford University Press, Inc., New York, 1996.

[30] C. Fragouli, J. L. Boudec and J. Widmer, "Network coding: an instant primer," *Computer Communication Review*, vol. 36, pp. 63–68, 2006.

[31] S. Yang and R. W. Yeung, "Large file transmission in network-coded networks with packet loss: a performance perspective," *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*, pp. 117:1–117:5, 2011.

[32] D. S. Lun, M. Medard, R. Koetter and M. Effros, "On Coding for Reliable Communication over Packet Networks," *Computing Research Repository (CoRR)*, vol. abs/cs/0510070, 2005.

[33] H. Wang, S. Xiao and C. J. Kuo, "Random linear network coding with ladder-shaped global coding

matrix for robust video transmission," *Journal of Visual Communication and Image Representation*, vol. 22, pp. 203–212, 2011.

[34] Y. Wang, S. Jain, M. Martonosi and K. Fall, "Erasure-coding based routing for opportunistic networks," *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pp. 229–236, 2005.

[35] M. Mitzenmacher, "Digital Fountains: A Survey and Look Forward ," *Proceedings of the IEEE Information Theory Workshop*, pp. 271–276, 2004.

[36] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society of Industrial and Applied Mathematics*, vol. 8, pp. 300–304, 1960.

[37] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman and V. Stemann, "Practical loss-resilient codes," *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pp. 150–159, 1997.

[38] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 569–584, 2001.

[39] P. A. Chou and Y. Wu, "Network Coding for the Internet and Wireless Networks," *Signal Processing Magazine, IEEE*, vol. 24, pp. 77–85, 2007.

[40] D. Y. Hu, M. Z. Wang, F. C. M. Lau and Q. C. Peng, "On the design of low compexity decoding (LCD) network codes," *Proceedings of the IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, pp. 269–273, 2010.

[41] M. Wang and B. Li, "How Practical is Network Coding?," *Proceedings of the 14th International Workshop on Quality of Service, IWQoS*, pp. 274–278, 2006.

[42] M. Luby, "LT codes," *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 271–280, 2002.

[43] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 14, pp. 2551–2567, 2006.

[44] P. Pakzad, C. Fragouli and A. Shokrollahi, "Coding Schemes for Line Networks," *Computing Research Repository (CoRR)*, vol. abs/cs/0508124, 2005.

[45] M. Champel, K. Huguenin, A. Kermarrec and N. L. Scouarnec, "LT network codes: low complexity network codes," *Proceedings of the 5th international student workshop on Emerging networking experiments and technologies*, pp. 39–40, 2009.

[46] Y. Li, E. Soljanin and P. Spasojevic, "Effects of the Generation Size and Overlap on Throughput and Complexity in Randomized Linear Network Coding," *Computing Research Repository (CoRR)*, vol. abs/1011.3498, 2010.

[47] D. Silva, W. Zeng and F. R. Kschischang, "Sparse Network Coding with Overlapping Classes," *CoRR*, vol. abs/0905.2796, 2009.

[48] P. Sadeghi, D. Traskov and R. Koetter, "Adaptive network coding for broadcast channels," *Proceedings of the Workshop on Network Coding, Theory, and Applications (NetCod '09)*, pp. 80–85, 2009.

[49] H. Wang, "Network coded flooding," Master's Thesis, Dept. of Telecommunications, Delft University of Technology, 2009.

[50] S. von Solms, "Exploiting the implicit error correcting ability of networks that use random

network coding," Master's Thesis, Dept. of Engineering, North-West University, Potchefstroom, South Africa, 2009.

[51] L. R. Ford and D. R. Fulkerson, *Flows in Networks*. Princeton University Press, Princeton, New Jersey, 1962.

[52] B. V. Roy and K. Mason, "Lecture notes to the Introduction to Optimization," *Stanford University*, 2008.

[53] C. Godsil and G. Royle, *Algebraic Graph Theory*. Springer-Verlag New York Inc., New York, 2001.

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

[55] D. Silva, "Error Control for Network Coding," Ph.D. dissertation, Dept. Electrical and Computer Engineering, University of Toronto, 2009.

[56] T. Ho and D. Lun, *Network Coding: An Introduction*. Cambridge University Press, New York, 2008.

[57] S. Jaggi, et al., "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, vol. 51, pp. 1973–1982, 2005.

[58] J. Goseling, "Network Coding: Exploiting Broadcast and Superposition in Wireless Networks," Ph.D dissertation, Dept. of Telecommunications, Technische Universiteit Delft, 2010.

[59] Q. Wang, S. Jaggi and S. R. Li, "Binary Error Correcting Network Codes," *Computing Research Repository (CoRR)*, vol. abs/1108.2393, 2011.

[60] T. Ho, et al., "A Random Linear Network Coding Approach to Multicast," *IEEE Transactions on Information Theory*, vol. 52, pp. 4413–4430, 2006.

[61] O. Trullols-Cruces, J. M. Barcelo-Ordinas and M. Fiore, "Exact Decoding Probability Under Random Linear Network Coding," *IEEE Communications Letters*, vol. 15, pp. 67–69, 2011.

[62] D. Platz, D. H. Woldegebreal and H. Karl, "Random Network Coding in Wireless Sensor Networks: Energy Efficiency via Cross-layer Approach," *Proceedings of the IEEE 10th International Symposium on Spread Spectrum Techniques and Applications*, pp. 654–660, 2008.

[63] D. E. Lucani, M. Medard and M. Stojanovic, "Random linear network coding for time-division duplexing: field size considerations," *Proceedings of the 28th IEEE conference on Global telecommunications*, pp. 4601–4606, 2009.

[64] S. Vyetrenko, T. Ho and E. Erez, "On noncoherent correction of network errors and erasures with random locations," *Proceedings of the 2009 IEEE international conference on Symposium on Information Theory*, vol. 2, pp. 996–1000, 2009.

[65] L. Song, R. W. Yeung and N. Cai, "A separation theorem for single-source network coding," *IEEE Transactions on Information Theory*, vol. 52, pp. 1861–1871, 2006.

[66] R. Ahlswede and H. Aydinian, "On error control codes for random network coding," *Proceedings of the Workshop on Network Coding, Theory, and Applications (NetCod '09)*, pp. 68–73, 2009.

[67] C. Fragouli and E. Soljanin, "Network coding applications," *Foundations and Trends in Networking*, vol. 2, pp. 135–269, 2007.

[68] M. Sanna and E. Izquierdo, "A Survey of Linear Network Coding and Network Error Correction Code Constructions and Algorithms.," *International Journal of Digital Multimedia Broadcasting*, vol. 2011, pp. 1–12, 2011.

[69] T. Tirronen, "Optimizing the Degree Distribution of LT codes," Masters Thesis, Dept of Electrical and Communications Engineering, Helsinki University of Technology, 2006.

[70] F. D. Lima and Barros, "Topology matters in network coding," *Springer Telecommunications systems*, pp. 1–11, 2011.

[71] F. J. Böning, M. J. Grobler and A. S. J. Helberg, "Topological Arrangement of Nodes in Wireless Networks Suitable for the Implementation of Network Coding," *Proceedings of the Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*, p. 7, 2010.

[72] M. Jafari, L. Keller, C. Fragouli and K. Argyraki, "Compressed Network Coding Vectors," *Proceedings of IEEE International Symposium on Information Theory*, vol. 1, pp. 109–113, 2009.

[73] C. Fragouli, "Network Coding for Dynamically Changing Networks," *Wireless Communications and Mobile Computing Conference, 2008. IWCMC '08. International*, pp. 39–44, 2008.

[74] A. Hessler, T. Kakumaru, H. Perrey and D. Westhoff, "Data obfuscation with network coding.," *Computer Communications*, vol. 35, pp. 48–61, 2012.

[75] P. Cataldi, M. P. Shatarski, M. Grangetto and E. Magli, "Implementation and Performance Evaluation of LT and Raptor Codes for Multimedia Applications," *Proceedings of the 2006 International Conference on Intelligent Information Hiding and Multimedia*, pp. 263–266, 2006.

[76] R. W. Yeung, "On the minimum average distance of binary codes: linear programming approach," *Journal on Discrete Applied Mathematics*, pp. 263–281, 2001.

[77] S. A. Aly, V. Kapoor, J. Meng and A. Klappenecker, "Bounds on the Network Coding Capacity for Wireless Random Networks," *Computing Research Repository (CoRR)*, vol. abs/0710.5340, 2007.

[78] H. Wang, P. Fan and K. B. Letaief, "Maximum flow and network capacity of network coding for ad-hoc networks," *Trans. Wireless. Comm.*, vol. 6, pp. 4193–4198, 2007.

[79] D. Goldsman and G. Tokol, "Output analysis: output analysis procedures for computer simulations," *Proceedings of the 32nd conference on Winter simulation*, pp. 39–45, 2000.

[80] W. Press, S. Teukolsky, W. Vetterling and B. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed., Cambridge University Press, New York, 2007.

[81] J. Lambers, "Lecture notes to Numerical Linear Algebra," *Department of Mathematics, University of Southern Mississippi*, 2011.

[82] B. Shrader and N. M. Jones, "Systematic wireless network coding," *Proceedings of the Military Communications Conference MILCOM*, pp. 1–7, 2009.

[83] D. Heckerman, "A Bayesian Approach to Learning Causal Networks," *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pp. 285–295, 1995.

Appendix A

The results obtained and presented in this thesis have been published in the following journal and conference papers.

J1: S. von Solms and A.S.J. Helberg, “An evaluation of redundancy for Implicit Error Detection in Random Network Coding,” Errata to C1, improved and extended version of C1 for possible future submission.

J2: S. von Solms and A.S.J. Helberg, “Random Linear Network Coding for Belief Propagation Decoding,“ submitted to the Journal of Network and Computer Applications, Sept 2012.

J3: S. von Solms and A.S.J. Helberg, “Modified Earliest Decoding in networks that implement Random Linear Network Coding,“ Africa Research Journal, vol. 103, no. 4, pp. 165–171, Dec 2012.

J4: S. von Solms and A.S.J. Helberg, *“Evaluation of Decoding Methods for Random Linear Network codes*,“ submitted to the Journal of Network and Computer Applications, Nov 2012.

C1: S. Von Solms, S. J. de Wet and A. S. J. Helberg: “Error correction for resource limited random network coding networks,” Proceedings of the IEEE Africon, Livingstone, Zambia, September 2011.

C2: S. von Solms and A.S.J. Helberg, “The implementation of LT network coding in resource limited RLNC networks,” Proceedings of the Southern Africa Telecommunication Networks and Applications Conference (SATNAC), East London, South Africa, 2011

C3: S. von Solms and A.S.J. Helberg, “Encoding for belief propagation decoding in random network codes,” Proceedings of the Southern Africa Telecommunication Networks and Applications Conference (SATNAC), Fancourt, George, South Africa, 2012.

C4: S. von Solms and A.S.J. Helberg, “Modified Earliest Decoding for Random Network Codes,” Proceedings of the 2011 International Symposium on Network Coding (NetCod), Beijing, China, 2011.

Table A.1 shows where the contributions of the journal or conference paper can be found in this thesis.

Table A.1: Journal and conference contributions

	Chapter 4	Chapter 5	Chapter 6	Chapter 7
J1	x			
J2		x		
J3			x	
J4			x	x
C1	x			
C2		x		
C3		x		
C4			x	

Error correction for resource limited random network coding networks

Suné von Solms, Sarel J. de Wet, Albert S. J. Helberg
School for Electric, Electronic and Computer Engineering
North-West University, Potchefstroom Campus
Potchefstroom, South Africa
E- mail: sune.vonsolms@nwu.ac.za, joubert.dewet@nwu.ac.za, albert.helberg@nwu.ac.za

Abstract— **Random linear network coding is a practical approach to network coding. It is, however, susceptible to corruption of the message packets due to hostile factors in the network. Error correction can be implemented, but in some resource limited networks nodes cannot afford to transmit additional parity packets for error correction. In [1-3] a method is presented where error correction can be implemented at the receiver nodes of the network without the transmission of parity packets over the network. The parity required for error correction is obtained from redundant packets collected by receiver nodes. In this paper we extend the method for the use in resource limited networks due to the reduction in network overhead and transmission resources as well as the increase of coding opportunities.**

Keywords- Linear Error Correction; Network Coding; Random Linear Network Coding

I. INTRODUCTION

The field of random linear network coding (RLNC) and the advantages it offers in wireless networks, such as sensor networks, has been extensively studied in the past years; see [4-8]. RLNC allows a more practical approach to network coding, where there is no need for centralized network control and planning. This leads to an improvement in network throughput as well as energy efficiency and delay [9].

In a RLNC network, a source node transmits k information packets over the network. The intermediate network nodes create a linearly encoded packet from the packets received on their incoming edges which is then transmitted. The receiver nodes collect at least k network coded packets from the network in order to decode the transmitted information. This allows the receiver node to decode the transmitted data upon reception of any set of random encoded packets of sufficient rank, where the information regarding the source packets included in each received packet are described by the coding vector included in the header of the packet [4,10].

RLNC environments, however, are subjected to a variety of hostile factors like packet-losses, link failures, noise, an insufficient network min-cut and the occurrence of errors caused by malicious or malfunctioning nodes. Due to these factors, reliable networks must be designed to be capable of countering the effects of such errors. These requirements are widely addressed by implementing error correction in RLNC networks. A non-deterministic approach to network error correction is addressed and studied in [5-8]. The network error correction method requires the source nodes of the network to encode the information packets by adding parity and is described fully in [5].

The implementation of a forward error correction code at the source node of the network encodes the information packets into coded packets that are transmitted. This encoding method must be known to the receiver nodes in order to successfully implement the correct error correction scheme. The information regarding the chosen error correction scheme must be communicated by the source node to the respective receivers.

In this paper we analyze the implementation of the method suggested in [1-3] in a resource limited network. This method does not require the encoding of information packets at the source nodes of the network, but utilizes redundant information at the receiver nodes for error correction.

II. MOTIVATION

Next we consider the advantages of transmitting less source packets into resource limited networks.

The transmission of an encoding vector in the header of a source packet causes additional overhead. In networks where large packets are transmitted, the coding vector is small relative to the data and has no significant influence on the packet overhead. In wireless sensor networks, the source packets only consist of a few bits. Appending an encoding vector to those packets has a severe influence on the packet overhead [11,12]. Thus, transmitting k source packets without encoding it at the source reduces the size of the coding vector. This reduction in packet overhead has an influence when used in networks like a wireless sensor network.

The transmission of a sequence of k source packets, instead of n coded packets reduces the transmissions of the source node. The intermediate network nodes require fewer resources due to random calculations performed on k source packets instead of $n > k$. Due to the transmission less source packets, the number of packets required to be stored at an intermediate node is also reduced.

The reduction in buffer size and transmissions sequences leads to a more favorable environment for RLNC to be implemented. According to a study by [13-15] the coding opportunities in a wireless network is better when the transmitted packets are smaller.

III. BACKGROUND

Firstly, we present the model for a RLNC network as well as a few linear error correction concepts core to our problem.

A. Model

We adopt the notation used in [4] for an acyclic network model. Consider a RLNC network as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The network consists of source node $S \in \mathcal{V}$ and a set of receiver nodes $T = \{t_1, \dots, t_d\}, T \in \mathcal{V}$. The set of edges $\mathcal{E}$ represents the communication channels, and there are $|\mathcal{V}|$ nodes in the network.

A source node, S, contains k information packets $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k\}$ of length m from the finite field $\mathbb{F}_{2^m}$. These source packets are transmitted over a random linear network coding network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The intermediate network nodes generate random linear encoded packets from the packets received on their incoming links. These packets are transmitted on the outgoing edges, eventually reaching the receiver nodes.

For each receiver node $\tau \in T$ to decode the transmitted message, it collects $k' \geq k$ encoded packets from the network in the form

$$y_j = \sum_{i=1}^{k} q_{ij} x_i \,, j = 1,2, \dots k' \tag{1}$$

where the coefficients $\{q_{ij}\}$ are randomly generated from a finite field $\mathbb{F}_{2^q}$.

The receiver packets, however, may be corrupted due to hostile factors mentioned earlier. In order to counter the effect of possible errors in the network, we use the additional collected packets at the receiver nodes for error correction. We need to select $(n-k)$ valid parity packets from the obtained packets in order to successfully correct possible errors. Effectively, we encode the k transmitted information packets into a code word of n coded packets using a linear (n, k, d) error correction code.

B. Linear error correction

Linear error correction is a well known field and studied in a range of textbooks, including [16]. We present the basic concepts that are core to our problem.

A block code converts a sequence k source packets, into a transmitted sequence of n packets, where $k < n$. For a linear block code C, the additional $d = (n-k)$ packets are linear functions of the original k packets, as defines by the generator matrix, $\boldsymbol{G}$, of code C. [17]. A linear code C has a minimum distance $d_{min} = 2t + 1$, then the code can correct t errors or detect $2t$ errors.

It is possible for a linear code C to have several distinct generator matrices, but not all of the $k \times n$ possible matrices are valid generator matrices. It is proven in [18] that a $k \times n$ matrix is a generator matrix if and only if it has a rank of k (k linearly independent rows) and its row vectors $\{\boldsymbol{g_i}\}$ are valid code words in code C.

The general property of a valid generator matrix that is of interest to us is the following: a generator matrix $\boldsymbol{G}$ are composed from two sub matrices $\boldsymbol{K}$ and $\boldsymbol{P}$; where $\boldsymbol{K}$ is a $k \times k$ matrix of rank k and $\boldsymbol{P}$ is a $k \times (n-k)$ matrix of rank $(n-k)$. The combination of $\boldsymbol{K}$ and $\boldsymbol{P}$ form a valid generator matrix which renders valid codes that are permuted through matrix row operations.

For any $k \times n$ matrix $\boldsymbol{G}$ over finite field $\mathbb{F}_{2^q}$ with k linearly independent rows, there exists a single $(n-k) \times n$ matrix $\boldsymbol{H}$ over $\mathbb{F}_{2^q}$, where

$$\boldsymbol{G} \cdot \boldsymbol{H}^T = \boldsymbol{0} \tag{2}$$

The parity check matrix $\boldsymbol{H}$ has $(n-k)$ linearly independent rows and describes the minimum distance of the linear code C: code C has a minimum distance d_{min} when d_{min} is the smallest number of columns of H which sum equals the zero vector [16].

These above properties of $\boldsymbol{G}$ and $\boldsymbol{H}$ matrices are used in section IV by receiver nodes to correct possible errors.

IV. CONSTRUCTION OF GENERATOR MATRIX AT RECEIVER NODES

The encoding of code words traditionally takes place at the network source node and then transmitted over the network. In a RLNC network of sufficient min-cut $\geq n$, it is possible to only transmit the k data messages over the network and still be able to correct possible errors. RLNC allows for nodes to create linear combinations of the k source packets to be collected by the receivers. These packets can be used for error correction at the receiver nodes.

We consider a network with min-cut $\geq n$ where the source node only transmits the k data messages $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k\}$ over the network. The receiver node $\tau \in T$ collects $k' \geq n > k$ channel packets in the form:

$$y_j = \sum_{i=1}^{k} q_{ij} x_i \,, j = 1,2, \dots k' \tag{3}$$

where the coefficients $\{q_{ij}\}$ are the encoding vectors of the received packets that form a $k \times k'$ matrix $\boldsymbol{Q}$, where

$$\boldsymbol{Q} = \begin{pmatrix} q_{11} & q_{12} & \cdots & \cdots & q_{1k'} \\ q_{21} & q_{22} & \cdots & \cdots & q_{2k'} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ q_{k1} & q_{k2} & \cdots & \cdots & q_{kk'} \end{pmatrix} \tag{4}$$

The encoding vectors of each encoded packet are transmitted along with the packet in the message header [10]. Network properties (such as connectivity and min-cut) influence the combinations and number of the vectors in $\boldsymbol{Q}$. These encoding vectors captured in $\boldsymbol{Q}$ are evaluated and used to construct a valid generator matrix. The construction of $\boldsymbol{G}$ takes place in two steps:

1) Construction of sub matrix $\boldsymbol{K}$*:*

From matrix $\boldsymbol{Q}$ each receiver first collects k packets with linearly independent encoding vectors $\{q_i\}$, which form the column vectors of the $k \times k$ matrix $\boldsymbol{K}$. Due to the linearly independency of these vectors, they ensure that the rank of $\boldsymbol{K}$ is equal to k.

2) Construction of sub matrix $\boldsymbol{P}$*:*

From the remaining packets in $\boldsymbol{Q}$ each receiver collects another $(n-k)$ packets, where their encoding vectors $\{\boldsymbol{q_j}\}$ form the column vectors of the $k \times (n-k)$ matrix $\boldsymbol{P}$ with a rank equal to $(n-k)$.

These two sub matrices form a valid generator matrix $\boldsymbol{G}$:

$$\boldsymbol{G} = [\boldsymbol{K}\,\boldsymbol{P}]. \tag{5}$$

Each receiver uses the message packets y_j corresponding to the encoding vectors $\{q_j\}$ selected for $\boldsymbol{G}$ to construct the matching codeword, c. The receiver node can then use $\boldsymbol{G}$ to correct or detect errors up to the capability of $\boldsymbol{G}$.

In the following paragraph we provide an example to illustrate the construction of the $\boldsymbol{G}$ matrix. Note that in this example each coding vector are in $\mathbb{F}_2$, but can be extended to $\mathbb{F}_{2^q}$.

Example 1: Hamming (7,4) code.

Assume a source node, S, transmits a sequence $\boldsymbol{X}$ of k source symbols in $\mathbb{F}_2$ over a network with min-cut $\geq n$, where $\boldsymbol{X} = (1\,0\,1\,1)$. The symbols are network coded by the intermediate network nodes. One of the receiver nodes $\tau \in T$ collects a sequence $\boldsymbol{Y}$ of 8 network coded packets from the network,

$$\begin{aligned}\boldsymbol{Y} &= \boldsymbol{X} \cdot \boldsymbol{Q} \\ &= (1\,0\,1\,1) \cdot \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \\ &= (1\,0\,1\,0\,1\,0\,1\,1)\end{aligned} \tag{6}$$

The receiver node evaluates the encoding vectors in $\boldsymbol{Q}$ to construct sub matrices $\boldsymbol{K}$ and $\boldsymbol{P}$. For the construction of $\boldsymbol{K}$, the receiver needs to find $k = 4$ packets with linearly independent coding vectors. One possibility is

$$\boldsymbol{K} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}. \tag{7}$$

$\boldsymbol{K}$ is sufficient to decode the transmitted sequence, but not for error correction. For the construction of $\boldsymbol{P}$, the receiver needs to find $(n-k) = 3$ packets with linearly independent coding vectors.

$$\boldsymbol{P} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \tag{8}$$

These two sub matrixes are used to form a valid generator matrix $\boldsymbol{G}$**,** where

$$\boldsymbol{G} = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}, \tag{9}$$

the corresponding code word is $c = (1\,0\,1\,0\,1\,0\,1)$ and the parity check matrix for $\boldsymbol{G}$ is

$$\boldsymbol{H} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}. \tag{10}$$

We can now calculate the syndrome $z = c \cdot \boldsymbol{H}'$, to determine if error has occurred in the network and if it can be corrected. The error correction method is discussed in [16]. In a network with multiple receivers, each receiver node, $\tau \in T$, is able to construct a generator matrix and corresponding code word from the specific encoded packets they receive from the network.

The error correction and detection capability of the code relies on the structure of the generator matrix. When the minimum Hamming distance of the code words generated by $\boldsymbol{G}$ is $d_{min} = 2$, an error can be detected but not corrected.

V. Error correction and detection probability

In a RLNC network, successful decoding is not always guaranteed due to the non-deterministic characteristics of the network. In this section we evaluate the probability of receiving network coded packets from a RLNC network that we are able to use for error detection and correction.

We consider a RLNC of sufficient min-cut $\geq n$ where source packets are encoded randomly, independently and are non-zero. From [19], we assume that the non-zero encoded packets received from the network by the receiver nodes are distributed according to the Gaussian distribution. From these calculations, we follow the procedure described in [20].

In the method discussed in section IV the d_{min} of $\boldsymbol{G}$ relies on the structure of sub matrix $\boldsymbol{P}$. Firstly, we calculate the probability of collecting sufficient column vectors for $\boldsymbol{P}$ in order to generate a $\boldsymbol{G}$ that corresponds to a linear code of $d_{min} \geq 2$.

The probability of collecting the required k linear independent packets to construct a valid $\boldsymbol{K}$ matrix from the first k packets that we collect from the network equals:

$$\rho_k = \prod_{i=1}^{k} \left(\frac{2^k - 2^{i-1}}{2^k - 1}\right) \tag{11}$$

Next, we calculate the probability of collecting $(n-k)$ linearly independent packets for the construction of sub matrix $\boldsymbol{P}$. This probability equals:

$$\rho_{n-k} = \prod_{i=k+1}^{n} \left(\frac{2^k - k - 2^{i-k-1}}{2^k - 1}\right) \quad (12)$$

Thus, the probability of generating a valid $\boldsymbol{G}$ matrix from the first n collected packets is $\rho = \rho_k \times \rho_{n-k}$.

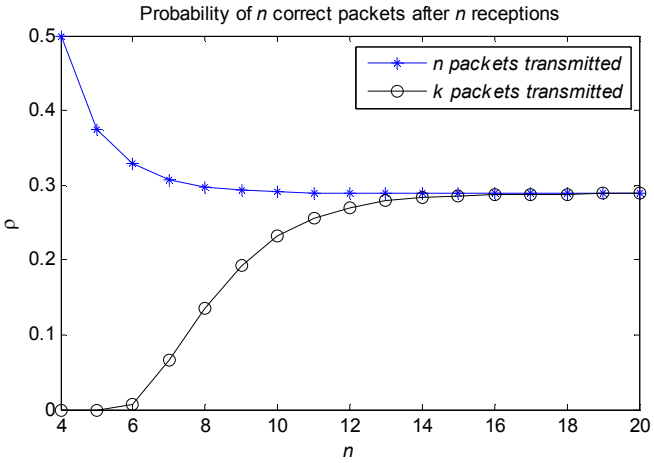


Figure 1: Probability of constructing valid $\boldsymbol{G}$ after n receptions

The probabilities for $n = 4,5,\ldots,20$ is plotted in Fig. 1. We compare this to the probability of receiving n linear independent packets which will be the case when n packets are transmitted by the source.

We can see that for large n, the probability ρ converges to approximately 0.2888, hence the probability of constructing a valid $\boldsymbol{G}$ matrix from the first n random packets collected is approximately 29% for both systems.

Next we calculate the number of expected random packets that must be collected by a receiver node in order to be ensured of n received packets that will generate a valid generator matrix. The probability distribution of the i randomly collected packets needed to ensure the successful collection of the required packets can be calculated through the use of a shifted geometric distribution $P(x = i) = \rho \times (1-\rho)^i, i > 0$.

The expected value is defined by:

$$E_k(x) + E_{n-k}(x) = \frac{1}{\rho_k} + \frac{1}{\rho_{n-k}} \quad (13)$$

where ρ_k and ρ_{n-k} is seen in (11) and (12) respectively. The following sum then gives us the total number of random collections of network packets a receiver node must make in order to construct a matrix $\boldsymbol{K}$ of rank k and matrix $\boldsymbol{P}$ of rank $(n-k)$:

$$\sum_{i=1}^{k} \frac{1}{\rho_k} + \sum_{j=k+1}^{n} \frac{1}{\rho_{n-k}} \quad (14)$$

Fig. 2 shows the number of additional packets expected to be collected by a receiver node in order to construct $\boldsymbol{G}$, i.e., $\sum(E_k(x) - k) + \sum(E_{n-k}(x) - (n-k))$ for $n = 4,5,\ldots,20$. We compare this to the additional packets expected by a receiver node to successfully collect n linearly independent packets when n packets are transmitted by the source.

It can be seen that the number of extra packets required converges to approximately 1.6 for a large n for both symbols. This means that we will be able to construct a valid and unique code word after approximately $n+2$ collected packets.

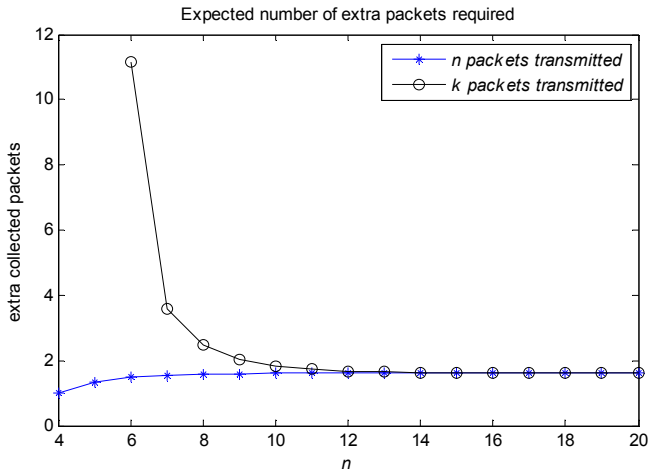


Figure 2: Expected number of extra packets required

Effectively, a network where n packets have been transmitted would be able to decode at the same point in time. The transmission of n packets at the source guarantees $d_{min} = 3$. However, when we construct a $\boldsymbol{G}$ matrix from sub matrices $\boldsymbol{K}$ and $\boldsymbol{P}$, we obtain a $d_{min} = 2$ and sometimes a $d_{min} = 3$. Thus, the selection of any $(n-k)$ linearly independent packets for $\boldsymbol{P}$ does not guarantee an error correction code C with $d_{min} \geq 3$.

In order to obtain such a single error correcting (n, k, d) linear code, one must construct a generator matrix $\boldsymbol{G}$ which encodes code words with Hamming distances $d_{min} \geq 3$.

The probability of collecting the required k linear independent packets to construct a valid $\boldsymbol{K}$ matrix from the first k packets that we collect from the network remains unchanged (11).

The probability of collecting $(n-k)$ packets for a $\boldsymbol{P}$ matrix that renders a $\boldsymbol{G}$ matrix with $d_{min} \geq 3$ is:

$$\rho'_{n-k} = \frac{c!}{d!\,(c-k)!} \Big/ \binom{2^k - 1}{d} \quad (15)$$

where $c = 2^d - 1 - d$ and d the minimal solution to the Hamming bound $2^d \geq d + k + 1$.

Thus the probability of collecting $\boldsymbol{K}$ and a corresponding $\boldsymbol{P}$ to render a maximum error correction $\boldsymbol{G}$ matrix is $\rho' = \rho_k \times \rho'_{n-k}$. The probabilities for $n = 7,8,\ldots,20$ is plotted in Fig. 3.

From the results it is clear to see that the probability of selecting random packets for the construction of a $\boldsymbol{G}$ matrix able to be used for error correction is very low.

From Fig. 3 and Fig.1 we can deduce that when a valid generator matrix is constructed from the collected network packets, the $\boldsymbol{G}$ matrix is more probable to only be able to detect a single error than to correct it.

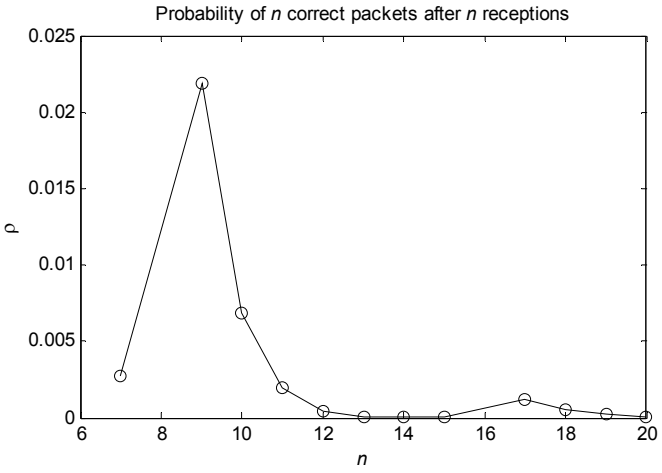


Figure 3: Probability of constructing valid G after n receptions for $d_{min} = 3$.

VI. CONCLUSION

We evaluated a technique where a code word is only constructed at the receiver of the network [1-3]. This method can be implemented opportunistically at each of the RLNC network receiver nodes. Because each network receiver node obtains different encoded packets due to the random encoding properties of the RLNC network, each receiver node can implement linear error correction based on the available channel packets.

This method is advantageous for networks where the size of transmissions must be kept as small as possible due to limited resources. The source packets transmitted over the network with this method are less which requires smaller buffers and yields a smaller network overhead. The transmission of k source packets instead of $n > k$ leads to more coding opportunities in wireless networks.

The probability of successful error correction at the receiver nodes is very low due to the non-deterministic characteristics of a RLNC network. The implementation of error detection, however, is high.

This method can be seen as an error correction/detection method that can be implemented opportunistically in resource scarce networks.

REFERENCES

[1] S von Solms, M.J. Grobler and A.S.J. Helberg, "Error Correction with the Implicit Encoding Capability of Random Network Coding,"Ad Hoc Networks: Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 2010, Volume 28, Part 1, 704-717.

[2] S. von Solms," Exploiting the implicit error correcting ability of networks that use random network coding," M.Eng Thesis, North-West University, School for Electric, Electronic and Computer Engineering, Nov 2009.

[3] S Von Solms and A. S J Helberg, "Performance of Implicit Error Correction of Network Coding networks in the presence of Link Errors," Proceedings of the 2009 Annual Conference of the South African Institute of Computer Scientists and Information Technologists, SAICSIT Conf. 2009, Vanderbijlpark, Emfuleni, South Africa, October 12-14, 2009.

[4] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in IEEE Int. Symp. Information Theory, Yokohama, July 2003, p. 442.

[15] Ahlswede, R. & Aydinian, H."On error control codes for random network coding" Network Coding, Theory, and Applications, 2009. NetCod '09. Workshop on, 2009, pp. 68-73.

[6] D. Silva, F. R. Kschischang, and R. Koetter, "A Rank-Metric approach to error control in random network coding," in: Proc. Information Theory for Wireless Networks, 2007 IEEE Information Theory Workshop, Solstrand, Norway, July 2007, pp. 1-5.

[7] N. Cai and R. W. Yeung, "Network error correction, part II: lower bounds, "Communications in Information Systems, vol. 6, no. 1, pp. 37-54, 2006.

[8] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," In: Proc. IEEE Transactions on Information Theory, Volume 54, Issue 8, August 2008, p. 3579 – 3591.

[9] H. Wang, J. Goseling, J.H. Weber, "Network coded flooding, " Master's thesis, Dept of Telecommunic., Delft University of Technology, June 2009.

[10] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in Allerton Conference on Communication, Control and Computing, Monticello, IL, October 2003.

[11] I.Broustis et al., First Report on Test-Bed Functionalities and Implementation of Network Coding Schemes", FP7-ICT-215252 NCRAVE "Network Coding for Robust Architectures in Volatile Environments", 30 July 2009, Revision Final, http://www.n-crave.eu.

[12] M. Jafari, L. Keller, C. Fragouli and K. Argyraki, "Compressed Network Coding Vectors," Proc. of IEEE International Symposium on Information Theory (ISIT 2009), Seoul, Korea, June 2009.

[13] F.J. Böning, M.J. Grobler and A.S.J. Helberg, "Topological Arrangement of Nodes in Wireless Networks Suitable for the Implementation of Network Coding", Proceedings of the Southern Africa Telecommunication Networks and Applications Conference (SATNAC), p. 7, Spier Estate, South Africa, 2010.

[14] F.J. Böning, "Topological arrangement of nodes in wireless networks suitable for the implementation of network coding", M.Eng Thesis, North-West University, School for Electric, Electronic and Computer Engineering, Nov 2010.

[15] LJ van Wyk, "Comparing network coding implementations on different OSI layers", M.Eng Thesis, North-West University, School for Electric, Electronic and Computer Engineering, Nov 2010.

[16] S. Lin and D. J. Costello, "Error control coding: Fundamentals and applications," Englewood Cliffs, N.J: Prentice-Hall, 1983.

[17] D. J.C. MacKay, "Information Theory, Inference, and Learning Algorithms", Cambridge University Press, 2003.

[18] O. Pretzel, Error-Correcting codes and Finite Fields," Oxford University Press Inc., NY, 1998.

[19] A. Hessler, T. Kakumaru, H. Perrey, D. Westhoff, "Data obfuscation with network coding," Computer Communications, Nov 2010.

[20] T.Tirronen, J. Virtamo, E. Hyytia, "Optimizing the Degree Distribution of LT codes," Master's Thesis, Helsinki University of Technology, Dept of Electrical and Communications Engineering, March 2006.

AN EVALUATION OF REDUNDANCY FOR IMPLICIT ERROR DETECTION IN RANDOM NETWORK CODING

S. von Solms and A.S.J. Helberg

School for Electric, Electronic and Computer Engineering, North-West University, Potchefstroom Campus, Potchefstroom, South Africa

E-mail: sune.vonsolms@nwu.ac.za, albert.helberg@nwu.ac.za

Abstract: A network that implements random linear network coding may be susceptible to corruption of the message packets. These errors are usually addressed through a concatenated forward error correction code implemented at the source and destination nodes. In this paper we present and evaluate an implicit error detection code where additional packets implicitly formed by the random linear network coding process are used to detect a single packet error. This scheme does not require the implementation of a forward error correction code at the source node. We evaluate this method by assessing the additional packets required by a receiver node for successful error detection. We present an analytical expression for the redundancy required for success and present simulation results to assess topology influence on this scheme. The obtained results show that with the collection of approximately 2 additional packets, a single error can be successfully detected.

1. INTRODUCTION

Random linear network codes (RLNC) was introduced in [1] and leads to an improvement in network throughput, energy efficiency as well as a reduction in delay [1-3]. Nodes do not need to carry knowledge of the topological network information or how the channel packets are encoded. The source node transmits n information packets over the network, where the intermediate network nodes create linearly encoded packets from the packets received on their incoming edges.

In a large network with high enough connectivity each receiver node collects $N > n$ network coded packets from the network, where N is slightly larger than n. Decoding can commence once the receiver has a set of random encoded packets of rank n, where the information regarding the source packets included in each received packet are described by the coding vector included in the header of the packet [3, 4].

Successful decoding, however, is subject to the receivers obtaining uncorrupted encoded packets. Due to hostile factors like packet-losses, link failures and noise error correction can be implemented to ensure transmission of reliable information. Different approaches to error correction in non-deterministic networks are presented in [5-8], e.g. implementation of forward error correction at the source node. These methods, however, lead to an increased load on the network.

We presented a method [9, 10] to detect a single error without the addition of parity packets at the source node or additional overhead in the network. This novel technique uses the packets implicitly formed by the random linear network coding process to construct a generator matrix for error detection.

In this paper we present an improvement to the method in [10] and evaluate it by deriving an analytical expression for the number of additional packets required to guarantee single error detection. We then conduct simulations to evaluate the influence of network topology on the scheme.

2. RELATED WORK

2.1 Random linear network coding

We adopt the notation used in [1, 11]. Consider an acyclic network which implements random linear network coding as a directed graph $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$. The set of edges $\mathcal{E}$ represents the communication channels, and there are $|\mathcal{V}|$ nodes in the network.

The network consists of a single source node $s \in \mathcal{V}$ and a set of receiver nodes $Z = \{z_1, \dots, z_{|z|}\}$, $Z \subset \mathcal{V}$. Let $r(s, Z)$ be the achievable rate at which s can multicast the source packets reliably to a set of receivers $Z \subset \mathcal{V}$. From the min-cut max-flow theorem, the value of min-cut(s, z) is the upper bound on $r(s, z)$ for any $z \in Z$ [12].

When min-cut $(s, z) \geq n$, the data present at s is divided into n packets to be multicast to Z. Assume $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n]$ are the n source packets where $\boldsymbol{x}_i$ represents the ith source packet from a finite field $\mathcal{F}$. These source packets are multicast over the edges $e \in \mathcal{E}$ of network $\boldsymbol{\mathcal{G}}$**.** At each intermediate network node v the packets received on its incoming edges e are randomly and linearly combined to form a new encoded packet to be transmitted on the outgoing edge e. An encoding vector is

included in the header of each outgoing packet and describes the source packets that have been linearly combined in the packet.

Each receiver node $z \in Z$ collects a set of $N \geq n$ encoded packets from the network, $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_N]$, where the jth encoded packet is of the form

$$\boldsymbol{y}_j = \sum_{i=1}^{n} q_{ij} \boldsymbol{x}_i \, , j = 1,2, \dots N \tag{1}$$

where the coefficients $\{q_{ij}\}$ are randomly generated from a finite field $\mathcal{F}_2$ and $\boldsymbol{q}_j$ forms the coding vector of packet $\boldsymbol{y}_j$. These coding vectors of the encoded packets can be represented as the column vectors of a $n \times N$ matrix $\boldsymbol{Q}$, where

$$\boldsymbol{Q} = \begin{pmatrix} q_{11} & q_{12} & \cdots & \dots & q_{1N} \\ q_{21} & q_{22} & \cdots & \dots & q_{2N} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ q_{n1} & q_{n2} & \dots & \dots & q_{nN} \end{pmatrix} \tag{2}$$

and $\boldsymbol{Y} = \boldsymbol{X} \times \boldsymbol{Q}$. The construction of $\boldsymbol{Q}$ is influenced by network properties such as connectivity and topology.

When N is slightly larger than n, there is a high probability that n encoding vectors stored in $\boldsymbol{Q}$ are linearly independent [1]. The receiver selects the n packets from $\boldsymbol{Q}$ that have linearly independent encoding vectors. The source packets are decoded by solving the linear system of equations through Gaussian elimination.

2.2 Network error correction

In network environments where errors may occur, forward error correction (FEC) codes are implemented with RLNC. The FEC codes are implemented as the outer code and RLNC as the inner code. This means that any receiver node can correct or detect possible errors using the FEC code after the random linear encoding of the network has been decoded [13].

Forward error correction entails that the data present at s is divided into k packets $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k]$ where $k < n$. Since a network with min-cut $(s, z) \geq n$ can support the independent transmission of n packets, the k source packets are mapped on n code packets $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_n]$.

In linear FEC, the n coded packets are linear functions of the original k source packets as defined by the columns of the $k \times n$ generator matrix $\boldsymbol{G}$ of the FEC code C [14, 15]

$$[\boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_n] = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k] \times \boldsymbol{G}. \tag{3}$$

For block codes there exist an important relationship between the block length n, dimension k and its error correcting capability, called the Hamming bound [14].

Definition 1: For any code $C = (n, k, d)$ with $d \leq 2e + 1$:

$$|C| \sum_{i=0}^{e} \binom{n}{i} \leq 2^n \tag{4}$$

where $d = 2e + 1$. A code is said to be perfect when there is equality in the bound. A perfect codes gives the optimal efficiency of an error correcting codes in relationship to the redundancy added.

This redundancy is used by z to determine if an error has occurred and if it can be corrected [14]. Following this encoding step, these encoded packets $\boldsymbol{U}$ are multicast over the edges $e \in \mathcal{E}$ of network $\boldsymbol{\mathcal{G}}$ to the receiver nodes in the process described in Section 2.1.

3. IMPLICIT ERROR DETECTION METHOD

In a network of sufficient min-cut where RLNC is implemented, error detection at the receiver nodes is possible without the addition of an outer code that adds parity [9, 10].

In this method s divides the source data into k packets $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k]$ where $k < n$, but the source packets are not encoded. These source packets are multicast over the edges e of network $\boldsymbol{\mathcal{G}}$ where the intermediate network nodes v perform RLNC to encode the packets.

In a network with min-cut $(s, z) \geq n$, the network can support the independent transmission of n packets and thus the random encoding characteristic of RLNC allows for the inherent production of n coded packets independently obtained from s. Obtaining parity from the network eliminates the need to construct and transmit parity at the source for error detection.

In this method, each receiver node $z \in Z$ collects a set of $N \geq n > k$ encoded packets $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_N]$ from the network, where the jth encoded packet is of the form

$$\boldsymbol{y}_j = \sum_{i=1}^{k} q_{ij}\boldsymbol{x}_i \, , j = 1,2, \dots N \tag{5}$$

where the coefficients $\{q_{ij}\}$ are randomly generated from a finite field $\mathcal{F}_2$ and $\boldsymbol{q}_j$ forms the global encoding vector of packet $\boldsymbol{y}_j$. These global encoding vectors are represented as the column vectors of a $k \times N$ matrix $\boldsymbol{Q}$

$$\boldsymbol{Q} = \begin{pmatrix} q_{11} & q_{12} & \cdots & \dots & q_{1N} \\ q_{21} & q_{22} & \cdots & \dots & q_{2N} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ q_{k1} & q_{k2} & \dots & \dots & q_{kN} \end{pmatrix}. \tag{6}$$

From the $N \geq n > k$ collected packets, each receiver node evaluates the encoding vectors $\{\boldsymbol{q}_i\}$ and selects packets to construct a $k \times n$ generator matrix $\boldsymbol{G}$ which, in most cases, is non-systematic.

Each receiver uses the message packets $\boldsymbol{y}_j$ corresponding to the selected encoding vectors in $\boldsymbol{G}$ to construct the matching codeword, $\boldsymbol{c}$. Through the construction of a valid parity check matrix $\boldsymbol{H}$, where $\boldsymbol{G} \times \boldsymbol{H}^{\boldsymbol{T}} = \boldsymbol{0}$, the receiver node can detect errors up to the capability of linear code C through traditional linear error detection decoding [14]. This method translates to an encoding system where the RLNC performed in the network is not only an inner code, but the mechanism for the construction of redundancy.

In a network with multiple receivers, each receiver node, $z \in Z$, is able to construct a generator matrix and corresponding codeword from the encoded packets they receive from the network.

There exist several distinct generator matrices for the implementation of a linear error correction and detection code C on k source packets. A $k \times n$ matrix is a valid generator matrix $\boldsymbol{G}$ when:

- it has a rank of k, i.e. it has k linearly independent columns,
- its rows vectors are valid code words in code C.

The error correction capability of the code C is determined by the structure of $\boldsymbol{G}$. When the minimum Hamming distance of the code words generated by $\boldsymbol{G}$ is $d_{min} = 2t + 1$, the constructed code is able to correct t errors or detect $2t$ errors [14].

4. BEHAVIOUR OF IMPLICIT ERROR DETECTION

The non-deterministic characteristics of a network that implements RLNC do not always guarantee successful implicit error detection at the receiver nodes.

A mathematical model [16] was used to determine the probability of a receiver node to obtain n linearly independent packets within the first n packets received. The model considered a network where the non-zero encoded packets received from the network by the receiver nodes are Gaussian distributed. Using the same model, we derive an analytical expression to calculate the probability of a receiver node obtaining n packets that can be matched to a valid generator matrix.

We consider a network that implements RLNC of sufficient min-cut $(s, z) \geq n$ where non-zero packets are encoded randomly and independently. The error correction capability of the linear error correction code relies on the structure of $\boldsymbol{G}$. Firstly, we calculate the probability of collecting sufficient column vectors to construct a generator matrix that corresponds to a linear code of $d_{min} \geq 2$.

The characteristics of a valid generator matrix, where $d_{min} \geq 2$, are the following:

- two linearly independent sets, of size k and $(n - k)$ respectively, must be present,
- each source packet must be represented in both linearly independent sets.

Although it is possible for the second set to contain linearly independent packets without all the data symbols present, it would not satisfy the condition of $d_{min} \geq 2$.

4.1 Probability of success

Firstly, we determine the probability, P, of obtaining a valid generator matrix $\boldsymbol{G}$ in the first n packets collected by a receiver node. In order to derive the exact expression for P, we need to calculate the following probabilities:

The probability $\mathrm{p_{n,k}}$ of obtaining a full rank set (k linearly independent packets) within the first n packets collected was determined [17] to be:

$$p_{n,k} = \prod_{i=0}^{k-1} \left(1 - \frac{1}{2^{n-i}}\right) \text{ for } n > k. \tag{7}$$

Next we calculate p_d, the probability of the remaining $(n-k)$ packets being linearly independent:

$$p_d = \prod_{i=1}^{n-k}\left(\frac{2^k - 2^{i-1}}{2^k - 1}\right) \tag{8}$$

and p_s, the probability of the remaining $(n-k)$ packets containing all the source symbols:

$$p_s = 1 - \frac{\left[\binom{2^{k-1}-1}{n-k} \times k\right] - D}{\binom{2^k - 1}{n-k}}, \tag{9}$$

where

$$D = \sum_{i=3}^{k-2}(-1)^{k-i} \times \binom{2^i - 1}{n-k} \times \binom{k}{k-i}. \tag{10}$$

The probability that a valid generator matrix can be constructed from the first n collected packets is

$$P = p_{n,k} \times p_d \times p_s \tag{11}$$

and is depicted in Fig. 1 for varying values of n. Fig. 1 also contains results from Monte Carlo simulations for the method described in Section 3. The simulation randomly and independently generates n packets and evaluates them. The results obtained from the simulation match that of the analytical expression.

The presented method improves on the preliminary work presented in [10] by performing an exhaustive evaluation of all the n packets received to form a valid $\boldsymbol{G}$. Although this method is computationally more expensive, the results show that an error can be detected with high probability after n received packets, for a large n.

4.2 Expected number of additional packets

Section 4.1 showed the probability of constructing a valid generator matrix from the first n packets received. Another option is to collect $N > n$ packets until enough is received to construct a valid generator matrix $\boldsymbol{G}$. In [16] a calculation was done to determine the number of additional packets required to receive n linearly independent packets. In this section we analyse implicit error detection in a similar manner to determine the expected number of packets required to guarantee the construction of a valid generator matrix.

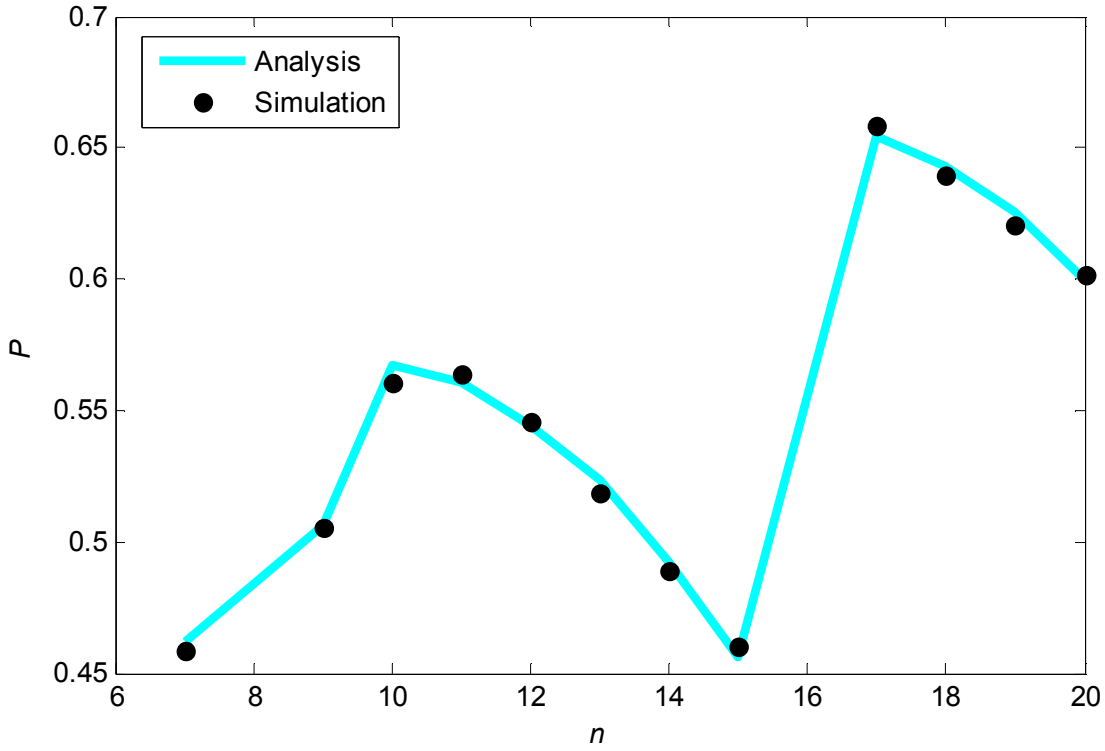


Figure 1: Probability of constructing a valid generator matrix after n received packets.

The probability of receiving i packets to obtain the next legitimate packet is geometrically distributed:

$$E_i = P \times (1-P)^{i-1}, i = 1,2 \ldots. \tag{12}$$

The expectation of (12) is equal to

$$\sum_{i=1}^{\infty} i \times E_i = \sum_{i=1}^{\infty} iP(1-P)^{i-1} = \frac{1}{P} \tag{13}$$

where P calculated in Section 4.1. The number of additional packets required to find the mth valid packet is:

$$\tau_m = \frac{1}{P_m}. \tag{14}$$

From this we can calculate the expected number of packets that will provide n valid packets for the construction of a generator matrix:

$$\sum_{m=1}^{n} \tau_m = \sum_{m=1}^{n} \frac{1}{P_m} \tag{15}$$

Fig. 2 shows $\sum_{m=1}^{n} \tau_m - n$, the number of additional packets required to successfully construct a generator matrix as well as the results obtained via Monte Carlo simulations.

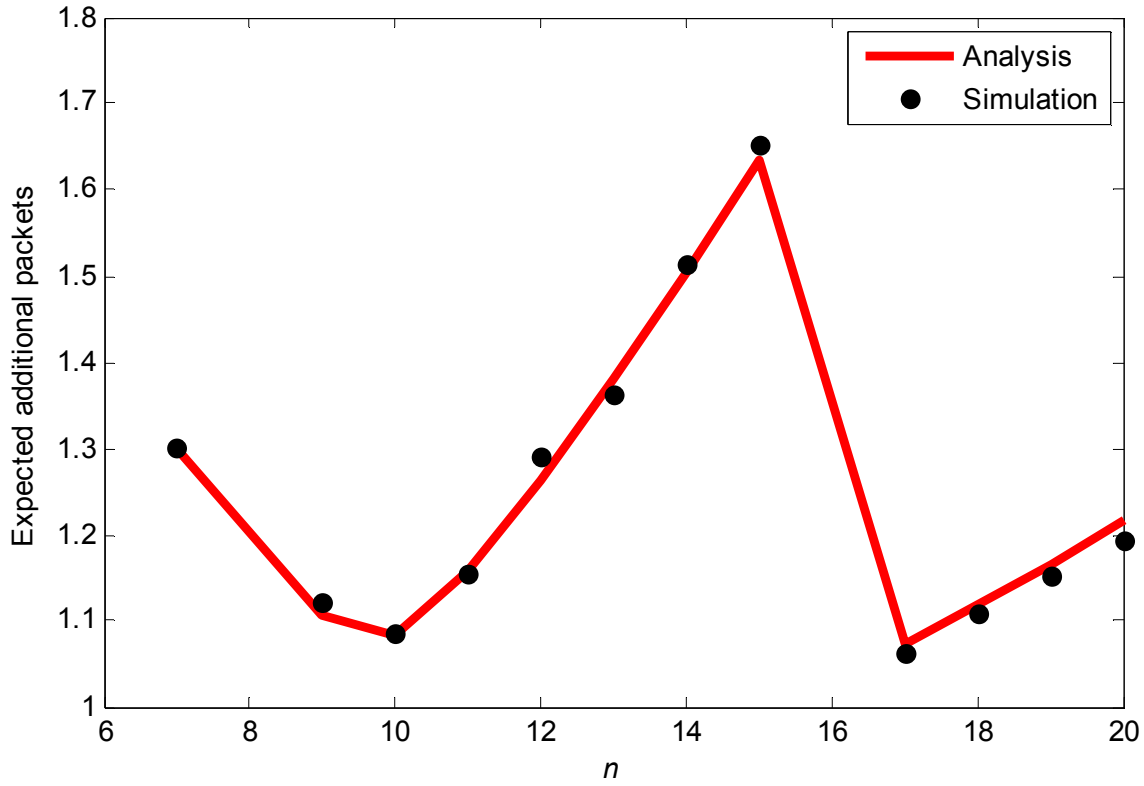


Figure 2: Expected number of additional packets required for the construction of a valid generator matrix.

It can be seen that the expected number of additional packets required for the construction of a $\boldsymbol{G}$ matrix that corresponds to a linear code of $d_{min} \geq 2$ is less than 2.

4.3 Discussion

It can be seen in Fig. 1 that the probability of constructing a valid $\boldsymbol{G}$ matrix for $n = 7$ and $n = 15$ dip to a minimum and thus maximises the number of additional packets required, shown in Fig. 2. As discussed in Section 2.2, forward error correction codes encodes k source packet into n coded packets using a predetermined algorithm. For the purpose of Implicit error correction the value of n

is determined by the min-cut of the network, i.e. the maximum number of packets that can be supported in the network to satisfy the condition of $d_{min} \geq 2$.

In certain cases, the codes formed by the receiver node are perfect codes that satisfy the equality

$$|C| \sum_{i=0}^{e} \binom{n}{i} = 2^n \tag{16}$$

These codes require a minimum number of redundant packets to satisfy the requirement of d_{min}, thus the probability of obtaining the minimum number of redundant packets are lower and the number of expected additional packets higher. There codes can be seen in Fig.1. and Fig.2. for $n = 7$ and $n = 15$.

5. SIMULATION SETUP AND RESULTS

In this section we conduct simulations to evaluate the influence of network topology on the scheme. In order to do so, we try to find the correlation between the analytical expression developed and the network environment.

We proceed to evaluate the mathematical model developed using Monte Carlo simulations. The mathematical model assumes that the packets collected by the receiver nodes are received uniformly at random and encoded independently. In large enough networks with high connectivity, the encoding at intermediate nodes and collection at the receiver nodes can be sufficiently modelled as such a random selection. In smaller less connected RLNC networks, however, this is not the case. Intermediate nodes have access to fewer packets and the encoded packets obtained at the receiver are not totally randomly generated.

We investigate the effect that network topology will have on the packets required to implement implicit error detection and consider two different network topologies.

5.1 Simulation setup

We base the experimental setup on that of [4] for an acyclic network model. The network is represented by graph $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes in the network and $\mathcal{E}$ the set of unit

edges in $\boldsymbol{\mathcal{G}}$ which represents the communication channels. We consider a randomly generated network with $|\mathcal{V}| = 100$ nodes and a single source s and receiver z for simplicity. The data to be transmitted by the source node are modelled as k source packets in the finite field $\mathcal{F}_2$. The min-cut of the network is min-cut $(s, z) \geq n$.

Two different network topologies are considered for this simulation to determine the influence of the network topology on the collection of packets. These topologies are based on that of [18].

- The Érdos-Rényi Graph, ER(100, δ): formed by randomly and independently assigning edges between all 100 nodes, so that each node has at least δ connected edges.
- The Random Geometric Graph, RGG(100, l): formed by placing 100 nodes uniformly at random on a unit square with communication radius of l.

The values for the RGG are specifically chosen so that the connectivity of the graphs is approximately the same as that of the ER graphs, with only a difference in topology. This allows us to make a direct comparison between the two different network models. Each simulation was performed 1000 times for both graphs and varying values of n.

5.2 Results

We evaluated the number of additional packets required by the receiver nodes in order to construct a valid $\boldsymbol{G}$ that corresponds to a linear code of $d_{min} \geq 2$ where k packets are transmitted by the source.

From Fig. 3 it can be seen that there is a significant difference between the results obtained for the RGG and ER graphs.

The ER graph: When the results in Fig. 3 are compared to the expected number of additional packets calculated in the analytical expression in Fig. 2, the values are comparable. In the ER graphs, nodes have an equal probability of connecting to any other node in the network. This allows information packets to be distributed randomly amongst all the nodes. Intermediate nodes may have access to a greater range of packets and the encoded packets obtained by the receiver node can be seen as a random selection of packets, which corresponds to the analytical expression.

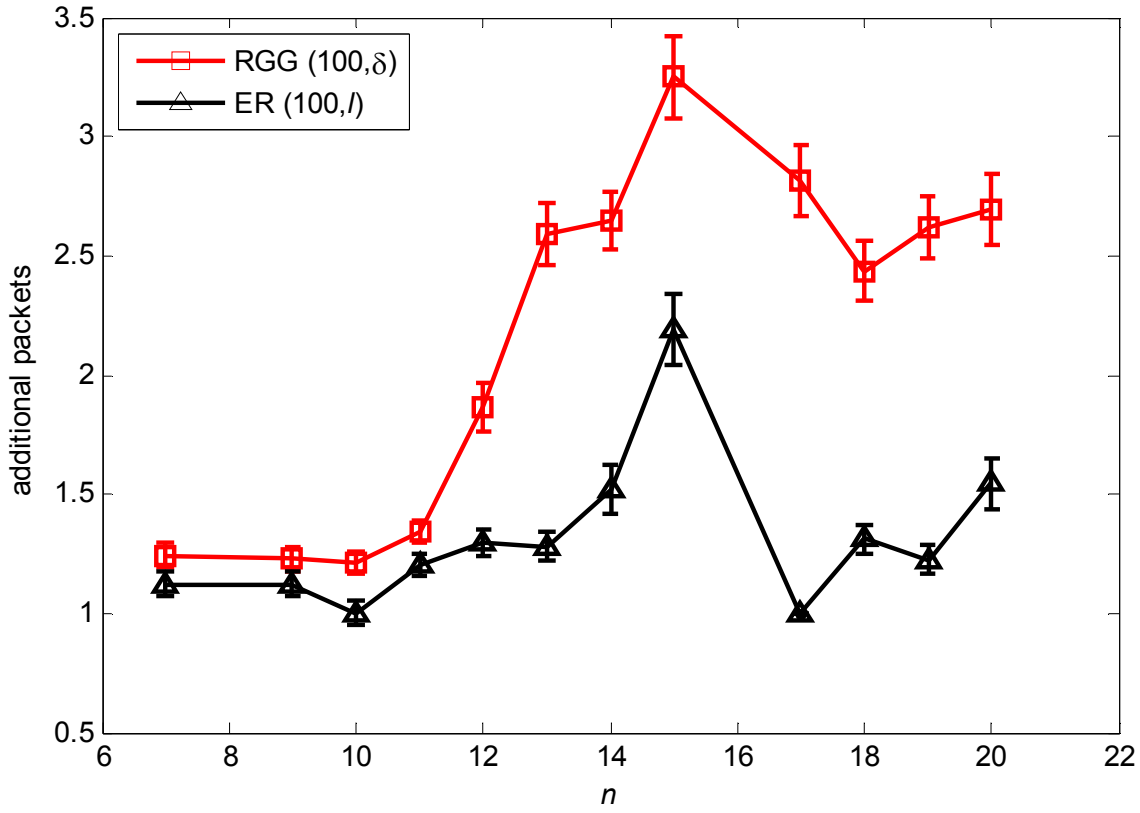


Figure 3: Expected number of additional packets required

The RGG graph: In a RGG graph the nodes only have edges to nodes within the range l. Thus packets in the network are distributed locally and intermediate nodes tend to encode only a restricted number of source packets. This results in more additional packets that have to be received for successful error detection. This corresponds to the basic principles of RLNC [1, 15].

6. THE CASE FOR $d_{min} \geq 3$

Forward error correction as applied at the source node s and decoded at the receiver node z requires approximately 2 additional packets for successful error correction [16].

We also determined the number of additional packets required to construct a $\boldsymbol{G}$ matrix that corresponds to a linear code which guarantees single error correction. In order to obtain such a single error correcting code, one must construct a generator matrix $\boldsymbol{G}$ which encodes code words with Hamming distances $d_{min} \geq 3$. We compare this result to the result acquired in [16]. The result is shown in Fig. 4.

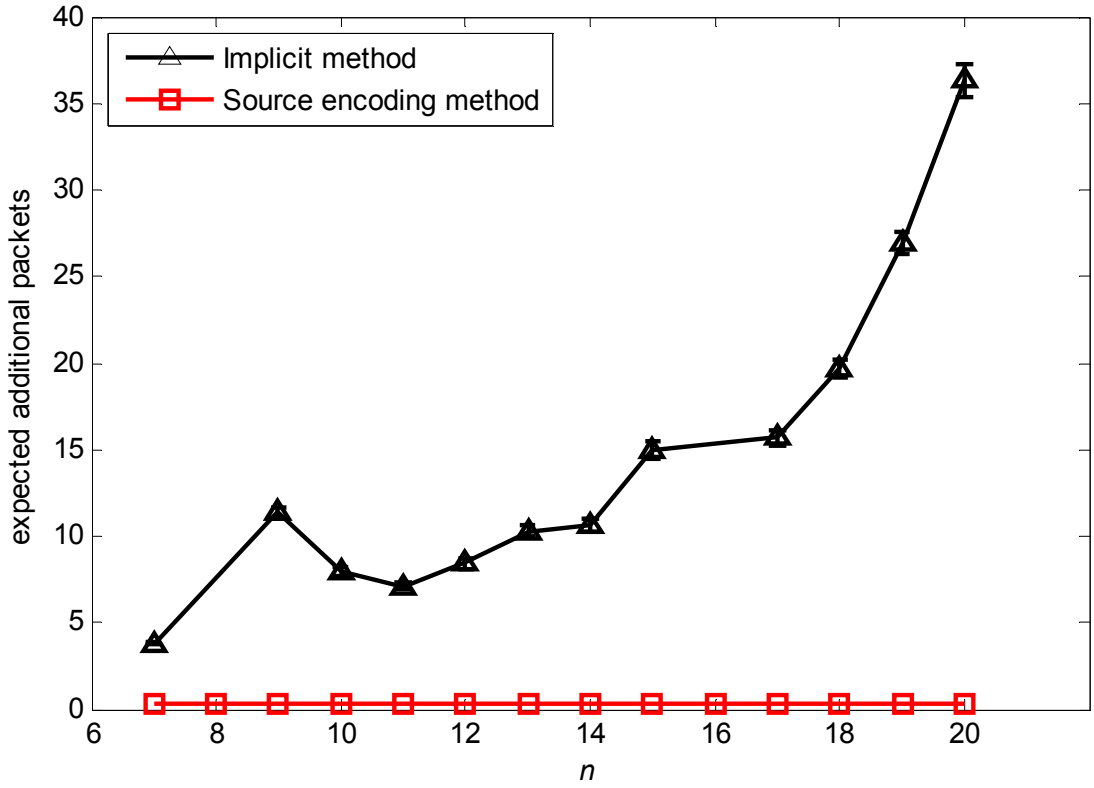


Figure 4: Number of extra packets required for $d_{min} = 3$ generator matrix.

It can be seen that the number of additional packets required for single error correction is very high and not practical.

7. CONCLUSION

In this paper, we improved and evaluated an implicit error detection technique from [10]. This method collects the packets implicitly formed by the random linear network coding process and constructs a generator matrix for error detection.

An analytical expression considering a network where the non-zero encoded packets received from the network by the receiver nodes are Gaussian distributed was constructed and validated. This model was used to:

- analyse the probability of constructing a valid $k \times n$ generator matrix after n received packets,
- calculate the number of additional packets required to construct a valid generator matrix.

The analytical expression showed that the reception of approximately 2 additional packets will guarantee the detection of a single error.

The analytical model was compared to the implementation of the implicit error detection method in two different network topologies. The number of additional packets required by a receiver node in the Érdos-Rényi network model was similar to the results obtained through mathematical analysis. The number of additional packets required by receiver nodes for the successful construction of a generator matrix in the Random Geometric Graph is higher than shown by the analytical expression.

The developed analytical expression describes the Érdos-Rényi graph accurately, because packets are more randomly distributed. The Random Geometric Graph is not accurately described by a network which receives randomly distributed encoded packets. The connections in the RGG network are more local and the packets obtained by a receiver node do not form a Gaussian distribution.

The implicit error detection method presented in this paper is advantageous for the use in networks with a large min-cut where the size of transmissions must be kept as small as possible due to limited resources, e.g. as in wireless sensor networks.

The source packets transmitted over the network with the implicit error detection method are shorter than packets where FEC codes are implemented at the source. This leads the use of less buffer space at the intermediate nodes, shorter switching time and shorter calculations due to fewer additions. The reduction in buffer sizes of intermediate nodes and smaller overhead in packets lead to a more favourable environment for RLNC [19].

A study performed by [19] shows that network coding opportunities in a wireless network are more favourable when the transmitted packets are smaller. In wireless network scenarios the transmission of smaller coding vectors in the packet header can be advantageous as information packets are small and may only consist of a few bits [20, 21].

8. REFERENCES

[1] Ho, T., Koetter, R., Médard, M., Karger, D. R., and Effros M.: "The benefits of coding over routing in a randomized setting". Proceedings: IEEE International Symposium on Information Theory (ISIT 2003), Yokohama, July 2003, pp. 442.

[2] Wang, H.: "Network coded flooding". Master's thesis, Department of Telecommunications, Delft University of Technology, Delft, 2009.

[3] Ho T., Medard, M., Koetter, R., Karger D.R., and Effros, M.: "A Random Linear Network Coding Approach to Multicast", IEEE Transactions on Information Theory, 2006, 52, (10), pp. 4413-4430.

[4] Chou, P. A., Wu, Y., and Jain, K.: "Practical network coding". Proceedings: Allerton Conference on Communication, Control and Computing, Monticello, IL, USA, October 2003, pp. 40-49.

[5] Ahlswede, R., and Aydinian, H.: "On error control codes for random network coding". Proceedings: Workshop on Network Coding, Theory, and Applications (NetCod), Lausanne, Switzerland, June 2009, pp. 68-73.

[6] Silva, D., Kschischang, F. R., and Koetter, R.: "A Rank-Metric approach to error control in random network coding". Proceedings: IEEE Information Theory Workshop: Information Theory for Wireless Networks, Solstrand, Norway, July 2007, pp. 1-5.

[7] Cai, N., and Yeung, R. W.: "Network error correction, part II: lower bounds", Communications in Information Systems, 2006, 6, (1), pp. 37-54.

[8] Koetter, R., and Kschischang, F. R.: "Coding for errors and erasures in random network coding", IEEE Transactions on Information Theory, 2008, 54, (8), pp. 3579-3591.

[9] Von Solms, S., Grobler, M.J., and Helberg, A.S.J.: "Error Correction with the Implicit Encoding Capability of Random Network Coding", in Zheng, J., Simplot-Ryl, D., ND Leung, V. C. M. (Eds.): "Ad Hoc Networks: Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering" (Springer, 2010, Vol. 28, Part 1), pp. 704-717.

[10] Von Solms, S., de Wet, S. J., and Helberg, A. S. J.: "Error correction for resource limited random network coding networks". Proceedings: IEEE Africon, Livingstone, Zambia, September 2011.

[11] Chou, P., and Wu, Y.: "Network Coding for the Internet and Wireless Networks", IEEE Signal Processing Magazine, 2007, 24, pp. 77-85.

[12] Fragouli, C., and Soljanin, E.: "Nework coding fundamentals", Foundations and trends in networking, 2007, 2, (1), pp 1-133.

[13] Cai, N., and Yeung, R. W.: “Network Coding and Error Correction”. Proceedings: IEEE Information Theory Workshop (ITW 2002), Bangalore, India, October 2002, pp. 119-122.

[14] Lin S., and Costello, D. J.: “Error control coding: Fundamentals and applications” (Prentice-Hall, 1983, 2nd edn. 2004).

[15] Pretzel, O.: “Error-Correcting codes and Finite Fields” (Oxford University Press Inc 1998).

[16] Tirronen, T.: “Optimizing the Degree Distribution of LT codes,” Master’s Thesis, Helsinki University of Technology, Dept of Electrical and Communications Engineering, 2006.

[17] Trullols-Cruces, O., Barcelo-Ordinas, J.M., and Fiore, M.: “Exact Decoding Probability Under Random Linear Network Coding”, IEEE Communications Letters, 2011, 15, (1), pp. 67-69.

[18] Lima, L., Ferreira, D., and Barros, J.: “Topology matters in network coding”, Springer Telecommunications systems, March 2011, pp. 1-11.

[19] Böning, F.J., Grobler, M.J., and Helberg, A.S.J.: “Topological Arrangement of Nodes in Wireless Networks Suitable for the Implementation of Network Coding”. Proceedings: Southern Africa Telecommunication Networks and Applications Conference (SATNAC), Spier Estate, South Africa, September 2010, pp. 7.

[20] Jafari, M., Keller, L., Fragouli C., and Argyraki, K.: “Compressed Network Coding Vectors”. Proceedings: IEEE International Symposium on Information Theory (ISIT 2009), Seoul, Korea, June 2009, pp. 109-113.

[21] Fragouli, C.: “Network Coding for Dynamically Changing Networks”. Proceedings: International Wireless Communications and Mobile Computing Conference (IWCMC '08), Crete Island, Greece, August 2008, pp. 39-44.

The implementation of LT network coding in resource limited RLNC networks

Suné von Solms, Albert S. J. Helberg
TeleNet Research Group
School for Electric, Electronic and Computer Engineering
North-West University, Potchefstroom Campus
Tel: (018) 299 1961, Fax: (018) 299 1977
E- mail: sune.vonsolms@nwu.ac.za, albert.helberg@nwu.ac.za

Abstract— We introduce an improved coding method constructed through the combination of the low complexity aspects of both random linear network coding (RLNC) and Luby Transform (LT) codes. The introduced method can successfully transmit data over a network while saving computational resources. When compared to the implementation of LT network coding, the proposed method consumes fewer computational resources at intermediate network nodes.

Index Terms— Belief propagation decoding, LT codes, Network Coding, Random Network Coding.

I. Introduction

Wireless devices play a vital part in the modern way of living and there have been many advances in the field of wireless communication. Challenges regarding the efficiency of information transmission remain due to interference from other sources, limitations in bandwidth, environmental factors, as well as limited available resources.

Computational resources at intermediate network nodes are traditionally consumed by the processing of received data to construct new data for transmission. Any repetitive or complex actions performed in the network may require nodes with larger computational resources like memory and computational power.

In this context, the implementation of random linear network coding (RLNC), introduced by Ho et al. [1] has proven to yield practical solutions for decentralized networks with limited resources [2, 3]. The implementation of RLNC in decentralised networks allows for the recoding of received information packets at intermediate nodes in the form of linear combinations of the received packets. This action leads to an improvement in network throughput as well as energy efficiency and delay [4].

Although the implementation of RLNC leads to an improvement in the utilisation of resources at the intermediate network nodes, this method requires a high complexity decoding process, called Gaussian elimination. Optimizations to decoding algorithms have been proposed, of which the Luby Transform (LT) codes [5] offers a significant improvement.

A method called LT network coding, proposed in [6], enables us to use a low complexity decoding scheme in a network that traditionally implements RLNC. This low complexity decoding scheme is based on belief propagation.

In order to implement this decoding scheme, the encoding process implemented at intermediate network nodes to encode new packets is more computationally complex than that of RLNC. Each network node must encode information packets according to a specified degree distribution, $\rho(d)$, which guarantees the efficient decoding through belief propagation at the receivers [5, 7, 8].

The implementation of LT network coding leads to a significant improvement in resource usage at the receiver nodes of the network. The disadvantage, however, is an increase in computational cost at each intermediate node.

In the context of limited resource networks, we present a method, called hybrid-LTNC, where we implement low complexity RLNC at the network nodes, except when it is connected to a receiver node. Network nodes connected to receivers implement the more complex LT network coding algorithm to allow for low complexity decoding.

II. Background

A. Random linear network coding

In the implementation of RLNC in networks, the data content are divided into k source packets $\{x_i\}_{i=1}^{k}$ of size m over a finite field $\mathbb{F}_q$. These source packets are transmitted from the source node to the intermediate nodes in the network.

The intermediate nodes implement RLNC where the coefficients for the linear combinations performed on the received packets are chosen uniformly at random. The encoding of a new packet $\{y_j\}_{j=1}^{k'}$ can be described by the linear combination of source packets,

$$y_j = \sum_{i=1}^{k} g_{ij} x_i \ , j = 1,2, \dots, k' \quad (1)$$

where $\{g_{ij}\}$ are the random coefficients from $\mathbb{F}_q$ forming a vector of size k, describing the source packets included. This vector is included in the header of the packet [6, 9].

This action is repeated until the packets are collected at the receiver nodes. Each receiver node, $\tau \in T$, collects $k' \geq k$ encoded packets that can be decoded if the selection is of sufficient rank. The use of Gaussian elimination for decoding requires $\mathcal{O}(k^3)$ operations where the increase in the number of source packets causes the complexity to grow significantly.

This high complexity decoding scheme is not ideal to implement at receiver nodes with access to limited resources, for it is both time and resource consuming. Thus the need for a less complex decoding method.

B. LT network coding [6]

LT network coding eliminates the need for Gaussian elimination at the receiver nodes through the implementation of low complexity belief propagation decoding. LT codes are a low complexity approach to linear coding, where the packets are encoded according to specified degree distribution, $\rho(d)$ [5]. A degree distribution specifies the number of source packets that must be included in each packet. For example: the degree of the encoded packet $y_i = \{x_1, x_2, x_4\}$ is $d_{y_i} = 3$. The optimal degree distribution is described by the Robust Soliton distribution introduced in [5] and shown in Fig. 1.

In the same manner as RLNC, the source node divides the data content into k source packets $\{x_i\}_{i=1}^{k}$ of size m from the finite field $\mathbb{F}_q$. Nodes encode packets where the linear combination of source packets is defined by the degree distribution.

The low complexity decoding algorithm of order $\mathcal{O}(m \cdot k \log k)$, is dependent on the statistical properties of the encoded packets used in the decoding process.

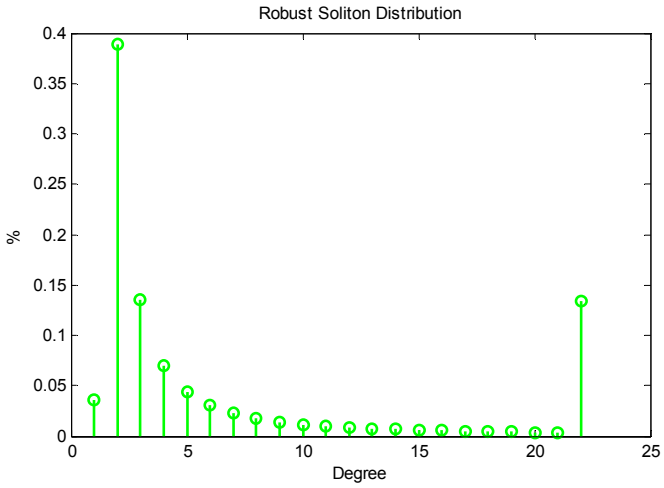


Figure 1: The Robust Soliton distribution for $k = 500$.

The decoding process can be described by the following algorithm [5]:

1. Find an encoded packet, $y_j, 1 \leq j \leq k'$, that only contains a single source packet, $x_i, 1 \leq i \leq k$.
2. Set source packet $x_i = y_j$ and delete y_j.
3. Subtract the value of x_i from all the other encoded packets $\{y_p\}_{p=1}^{k'}$ that contains source packet x_i.
4. Repeat from (1) until all source packets $x_i, i \in \{1, k\}$ are determined.

To ensure the success of the belief propagation decoding algorithm, the packets collected by the receiver nodes should adhere to this specific degree distribution. In order to achieve this, the intermediate nodes should not linearly combine the source packets randomly, but according to the Robust Soliton degree distribution.

To obtain the specified distribution requires more computationally complex encoding than that of RLNC. Intermediate nodes do not always have access to all the packets available at the source node, so building encoded packets according to a specific degree distribution proves to be resource intensive. This is due to the intermediate nodes which require that the encoding of a packet of a specific degree be built from the encoded packets they receive.

Note that each node receives random packets from the network including single-degree packets and packets of variable degrees containing a random collection of source packets. It is not guaranteed that all the source packets will be available at each network node and thus the new packet must be encoded from packets containing limited and arbitrary source packets.

In this paper, we reduce the resource usage in the network by only applying the LT network coding algorithm in [6] to intermediate nodes directly connected to receiver nodes. The other network nodes use RLNC to encode their packets. This hybrid-LTNC method decreases the computational requirements of the network which reduces the resource usage in the network.

In our study we design a simplified version of the algorithm proposed in [6], only looking at the degree-specific encoding of packets from the limited number of packets available at a node.

III. Model

From Section II we find that the belief propagation decoding algorithm relies on the statistical properties of the encoded packets collected at the network receiver nodes. We aim to deliver packets to receiver nodes that statistically match the Robust Soliton degree distribution, but only implement the specific encoding at nodes a single hop from the receiver nodes. The process applied at each intermediate network node is discussed subsequently.

An intermediate node establishes a connection with a neighboring node. The neighboring node then transmits a message via the feedback channel stating whether it is a receiver node or not. Based on the feedback information, the node then proceeds with the suitable encoding algorithm.

When the connection established by an intermediate node is not with a receiver, the node implements the low complexity RLNC for packet encoding. The encoding of a new packet using RLNC, as discussed briefly in section II, has been thoroughly presented by many researchers [1, 2, 9].

When the connection established by an intermediate node is with a receiver node, the following encoding procedure is followed to ensure that the receiver node receives packets according to the Robust Soliton degree distribution. This method is based on the method presented in [6]:

1) Selecting a target degree:

A target degree, d, is drawn from the Robust Soliton distribution. This target may not be reachable, meaning that no packet of degree d can be built from the packets present at the node. This is caused by the presence of the linear combinations of less than d source packets at the node:

$$\sum_{i=1}^{d} i \times n(i) < d \qquad (2)$$

where $n(i)$ is the number of encoded packets of degree i available at the node. If the selected target degree is determined as unattainable, a new target degree is selected until it can be reached.

2) Constructing packet of degree d:

This step builds a new encoded packet of degree d from a set of received packets so that the sum of their degrees equals d. The difficulty of this problem lies in the fact that the sum of the packets' degrees does not necessarily equal the degree of the sum of the packets. For example: a packet encoded from packets (x_1, x_2, x_4) and (x_3, x_4), renders a packet of degree 3 and not 5.

The node first examines the packets to determine if there already exists a packet of degree d. When packets of degree d are present, such a packet is selected as the new packet to be transmitted.

If no such packet is found, LT network coding implements a sub-optimal iterative process to construct a new packet of degree d. Firstly, the packets are evaluated in decreasing order of degrees, starting from $d-1$, where a packet is selected and named, t. Iteratively, packets are added to t, evaluating the resulting degree, d_t. When the addition of a packet increases d_t where $d_t \leq d$, the new packet is added to t. This process repeats until the target degree d is reached. When the target degree cannot be reached, the packet, t, with the closest degree to d is used.

IV. Evaluation

Next, we evaluate the performance by comparing the proposed hybrid-LTNC to the implementation of RLNC in the network and Gaussian elimination decoding at the receiver nodes; as well as complete LT network coding in the network nodes and belief propagation decoding at the receivers.

A. Experimental setup

We consider a randomly generated network with $N = 100$ nodes and $r = 20$ receivers. The data transmitted by the source node to all the receiver nodes consists of approximately 10000 packets in finite field $\mathcal{F}(2^8)$. These packets are divided into k transmission packets $\{x_i\}_{i=1}^{k}$ of size m from $\mathcal{F}(2^8)$. The connectivity of the network is $c \geq k$. This experimental setup is based on that of [9].

These packets are transmitted to the intermediate nodes in the network. When a node receives packets, it encodes a new information packet to transmit to connected nodes. We consider a multicast communication network and we assume a feedback channel allowing communication between nodes regarding connectivity to receiver nodes.

With the implementation of the three different schemes discussed, different algorithms were implemented at intermediate and receiver nodes, as summarized below:

1) Random linear network coding

Intermediate nodes encode a new data packet by randomly generating coefficients for the linear combinations performed on the received packets.

The receiver nodes implement decoding through Gaussian elimination on the received set of packets when of sufficient rank.

2) LT network coding

Intermediate nodes select a target degree from the Robust Soliton distribution. It encodes a new data packet by constructing it from the received packets as per the algorithm described in Section III.

The receiver nodes implement belief propagation decoding on the encoded packets received from the network.

3) Hybrid-LTNC

When a node is directly connected to a receiver node, the new packet is encoded according to a specific target degree from the Robust Soliton distribution. Nodes not directly connected to a receiver node implement RLNC to encode a new packet.

The receiver nodes implement belief propagation decoding on the packets received from the network.

B. Experimental results

Three different simulations were conducted to evaluate the trade-off between the performances of the different methods.

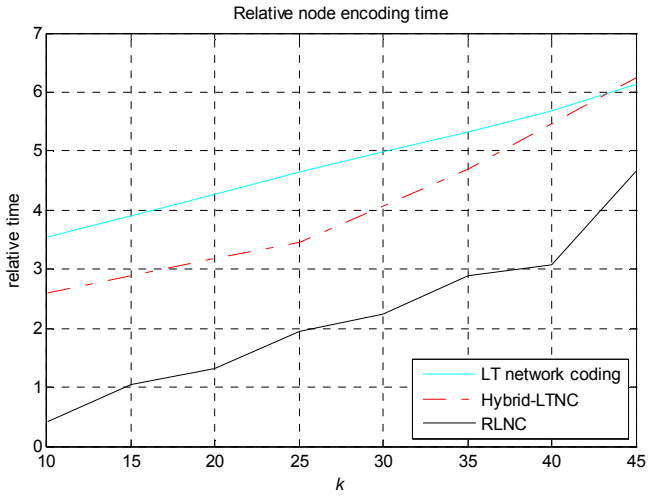


Figure 2: Relative packet encoding time per network node

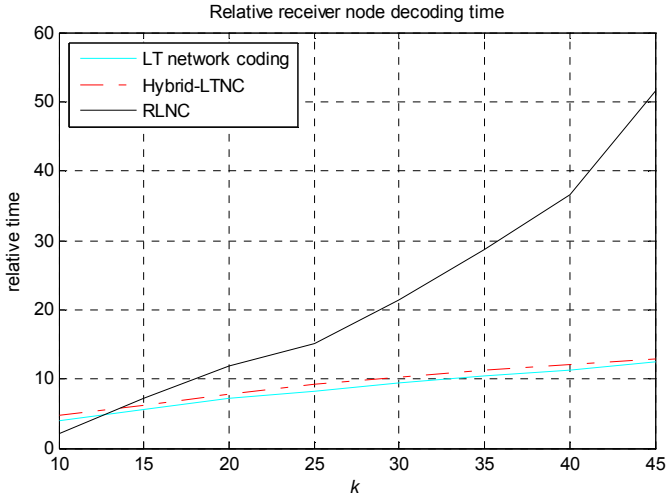


Figure 3: Relative decoding time per receiver node

For each simulation, we conducted 25 Monte-Carlo simulations. We conducted our simulations for values of k ranging from 10 to 45, varying the size of m accordingly.

1) Average packet encoding time per node

In this simulation, we compare the relative computational time at the intermediate nodes of the network for the different methods.

Fig. 2 depicts the average coding time per network node for a varying k. The average encoding time consumed by network nodes when implementing RLNC is shown as the solid dark line. The coding times per node for LT network coding and the proposed hybrid-LTNC method are represented by the solid light line and the broken line respectively.

It can be observed that the network resource consumption for the implementation of RLNC is the lowest. This is expected, since the process of RLNC has a very low complexity. The implementation of LT network coding at all the intermediate nodes is the most resource consuming due to the more complex algorithm run at every intermediate network node. The hybrid-LTNC method shows a smaller use of resource usage at the intermediate nodes. It is clear that the selected use of the LT network coding algorithm reduces the overall computational complexity of the network.

The complexity of all the algorithms increases as the number of source packets, k, increase. Although the size, m, of each block becomes smaller, the size of the coding vector increases. The process of constructing a new packet from received packets uses the coding vector in the calculations, resulting in an increasingly more complex method when the coding vectors are increased.

Also, the complexity of the Hybrid-LT method approaches that of the LT network coding method as k becomes large. As the number of k and the network connectivity increases, the number of intermediate nodes connected to receiver nodes increases.

2) Receiver decoding time

In this experiment, we analyse the average decoding time at the receiver nodes for Gaussian elimination and belief propagation decoding. Fig. 3 plots the average decoding time per receiver node for a varying value of k.

Fig. 3 confirms that the belief propagation decoding algorithm is less resource intensive than that of Gaussian elimination.

It can be observed that the resource consumption increases with the increase of k, although the size of m decreases. This can be confirmed by the order of complexity of this decoding algorithm, $\mathcal{O}(m \cdot k \log k)$. We can see that the number of source packets has a bigger impact on the decoding complexity than the size of the block.

Also, the receivers decode on average faster when all the network nodes implement LT network coding, instead of only those connected to the receiver nodes. The degrees selected from the Robust Soliton distribution are mostly very low, so when an intermediate node receives randomly encoded packets and then needs to encode a packet of a low degree, the target may not be reachable. This interferes with the statistical properties of the packets needed for decoding and hence increases the decoding time at the receiver nodes.

3) Additional packets

This simulation evaluates the number of additional packets required by each receiver node to successfully implement belief propagation for decoding. Fig. 4 plots the percentage of additional packets required for LT network coding or hybrid-LTNC in terms of varying k.

From Fig. 4 it can be seen that the number of additional packets collected by receivers for successful decoding is high, but reduces as k get larger; while the additional packets needed to implement Gaussian elimination is lower. Gaussian elimination only requires a set of packets of sufficient rank,

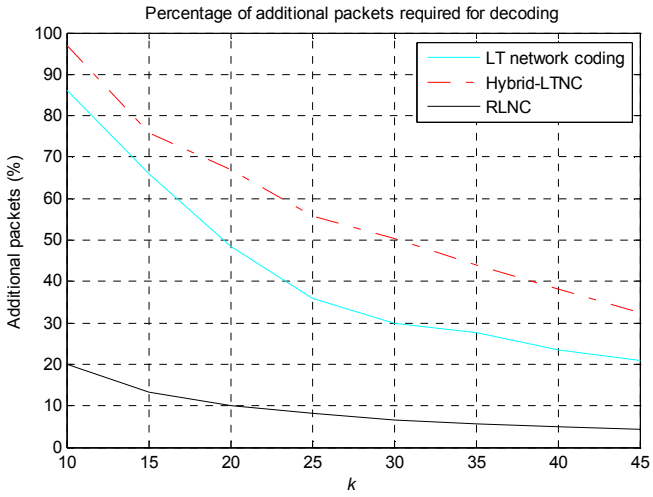


Figure 4: Percentage additional packets required at a receiver node

regardless of the structure of the source packets contained in them.

V. Conclusion

In this paper we conducted a comparative study of the hybrid-LTNC and LT network coding schemes. The former scheme is a combination of RLNC and LT network coding, extracting the low complexity advantages of both schemes. The implementation of this method in a RLNC network reduces the computational complexity of the network at the intermediate network nodes and allows for the implementation of a low complexity decoding scheme at the receiver nodes.

The disadvantage of this method is an increase in the additional packets needed to be collected by the receiver nodes before successful low complexity decoding can be completed. This in turn results in a longer decoding period at the receiver nodes.

Future work includes the optimization of the LT network coding algorithm to ensure accurate encoding of packets of the needed target degrees. This optimization will reduce the number of additional packets required for decoding which in turn will render faster decoding times.

VI. Acknowledgement

This work was completed with funding from the Telkom Centre of Excellence at the North-West University, Potchefstroom Campus.

VII. 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 *IEEE Int. Symp. Information Theory*, Yokohama, July 2003, p. 442.

[2] D. Silva, F. R. Kschischang, and R. Koetter, "A Rank-Metric approach to error control in random network coding," *Proc. Information Theory for Wireless Networks, 2007 IEEE Information Theory.*

[3] T. Ho, "Networking from a Network Coding perspective," PhD Thesis, Massachusetts Institute of Technology, Dept of EECS, May 2004.

[4] H. Wang, J. Goseling, J.H. Weber, "Network coded flooding, " Master's thesis, Dept of Telecommunic., Delft University of Technology, June 2009.

[5] M. Luby, "LT codes", Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on, 2002, pp. 271-280.

[6] M. Champel, K. Huguenin, A. Kermarrec, N. Le Scouarnec, "LT Network Codes", *Institut National de Recherch en Informatique et en Automatique (INRIA)*, November 2009.

[7] D. MacKay, "Fountain codes", Communications, IEE Proceedings-, 2005, vol. 152, pp. 1062-1068.

[8] D. J.C. MacKay, "Information Theory, Inference, and Learning Algorithms", Cambridge University Press, 2003.

[9] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," *Allerton Conference on Communication, Control and Computing*, Monticello, IL, October 2003.

Encoding for belief propagation decoding in random network codes

Suné von Solms, Albert S. J. Helberg
TeleNet Research Group
School of Electric, Electronic and Computer Engineering
North-West University, Potchefstroom Campus
Tel: (018) 299 1966, Fax: (018) 299 1977
E- mail: sune.vonsolms@nwu.ac.za, albert.helberg@nwu.ac.za

Abstract—The Hybrid- Luby Transform network code is an encoding method proposed for the implementation in communication networks employing random linear network coding. This method enables receiver nodes to implement low complexity belief propagation decoding. In this paper we show that the implementation of sparse random linear network coding and a less frequent buffer flushing policy to H-LTNC enables near optimal belief propagation decoding in a random linear network coding scenario.

Index Terms— Belief propagation decoding, Fountain codes, LT codes, Random Linear Network Coding.

I. Introduction

One of the criteria for measuring the effectiveness of a communication network is to determine its ability to transmit a bulk of data from the source to receiver nodes. There exist several challenges regarding effective information transmission in networks due to interference from other devices, environmental factors, as well as limited available resources.

In a network where the transmission between source and receiver nodes can be modelled by an erasure channel, fountain codes can be a very effective method of communication. Fountain codes, which are rateless codes, include Luby Transform (LT) codes [1] and Raptor codes [2]. Fountain codes require the source node to transmit n encoded source packets to the receiver via intermediate network nodes that only implement a store-and-forward algorithm to the packets they receive. A receiver is then able to decode the transmitted data when it receives N encoded packets, where $N = n + \delta$ with δ small in relation to n [1]. LT codes and Raptor codes require the source packets to be encoded according to a specific degree distribution as this distribution of packets allows for the implementation of a low complexity belief propagation (BP) decoding algorithm.

The store-and-forward technique implemented at the intermediate nodes of the communication network does not allow for the optimal utilisation of the communication channel. In order to utilise the communication channel optimally, Ho et al. [3] suggested the implementation of random linear network coding (RLNC). The implementation of linear coding at the intermediate nodes of the network leads to an improvement in the utilisation of network capacity which improves network throughput [4].

RLNC is a method that can easily be implemented in a practical network scenario by allowing intermediate nodes to randomly and linearly encode the packets received on their incoming edges to produce a new encoded packet. When the encoding at the intermediate nodes is done randomly and the operations are in a large enough finite field, the multicast capacity of the network can be reached [3].

The use of fountain codes in conjunction with RLNC in a communication network offers the advantage of low complexity BP decoding in a network that communicates at multicast capacity. LT codes require the encoding of packets to be according to the Robust Soliton (RS) degree distribution [1]. The random linear encoding at the intermediate network nodes, however, leads to *degree degeneration* where the specified input degree distribution degenerates with each random recoding at intermediate nodes in the network [5] so that the BP decoding at the receivers fail [6].

Several methods [6]-[8] have been presented to prevent the occurrence of degree degeneration at intermediate nodes allowing for the successful implementation of fountain codes in a RLNC environment. In this paper, we propose improvements on the method presented in [7] to allow linear encoding at most of the intermediate network nodes and low complexity BP decoding at the receivers.

II. Background

A. Random linear network coding

Consider an acyclic network $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$ implementing RLNC. The network consists of a single source node $s \in \mathcal{V}$ and a set of sink nodes $Z = \{z_1, \dots, z_{|z|}\}$, $Z \subset \mathcal{V}$. The achievable rate at which s can multicast the source packets reliably to the set of receivers Z is $r(s, Z)$. The maximum flow of the network for any $z \in Z$ is the upper bound on $r(s, z)$, thus min-cut $(s, z) \geq n$ [9].

The source data is divided into n packets, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n]$ where $\boldsymbol{x}_i$ represents the ith source packet of size m in a finite field $\mathbb{F}$ of size q. These source packets are

multicast over the edges $\mathcal{E}$ of the network from s. The intermediate nodes randomly and linearly combine, through X-OR operations, the packets received on their incoming edges e' to form a new encoded packet for transmission on their outgoing edges e. The encoding complexity at each intermediate node equals $\mathcal{O}(mb)$, where b is the size of the buffer [6]. In the header of each transmitted packet is a coding vector of length n, describing the included source packets $\boldsymbol{x} \subseteq \boldsymbol{X}$ [10].

Each receiver node $z \in Z$ collects a set of $N \geq n$ encoded packets $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N]$ from the network where the packets' global encoding vectors form the column vectors of a $n \times N$ matrix $\boldsymbol{G}$ where

$$\boldsymbol{X} \times \boldsymbol{G} = \boldsymbol{Y}. \quad (1)$$

The finite field $\mathbb{F}_q$ of the global encoding vectors is sufficiently large so that $\boldsymbol{G}$ is invertible with high probability when N is only slightly larger than n. The solution of the linear system of equations in (1) decodes the source packets $\boldsymbol{X}$ [10].

Traditionally the decoding method employed in RLNC networks is Gaussian Elimination whose matrix inversion algorithm is computationally complex and of order $\mathcal{O}(n^3 + mn^2)$ [6]. However, when the global encoding vectors in $\boldsymbol{G}$ resemble that of the RS degree distribution, low complexity BP decoding would be a more efficient decoding method.

B. Fountain codes

Fountain codes, which include LT codes, are a low complexity approach to linear coding. The optimal degree distribution for LT codes is described by the RS distribution. The encoding and decoding complexity of LT codes are $\mathcal{O}(mn \log n)$.

The BP decoding process can be described by the following algorithm [1]:

1. Find an encoded packet, $\boldsymbol{y}_j, 1 \leq j \leq N$, that only contains a single source packet, $\boldsymbol{x}_i, 1 \leq i \leq n$.
2. Set source packet $\boldsymbol{x}_i = \boldsymbol{y}_j$ and delete $\boldsymbol{y}_j$.
3. Subtract the value of $\boldsymbol{x}_i$ from all the other encoded packets $\{\boldsymbol{y}_p\}_{p=1}^{N}$ that contains source packet $\boldsymbol{x}_i$.
4. Repeat from (1) until all source packets $\boldsymbol{x}_i, 1 \leq i \leq n$ are determined.

C. RLNC and Fountain codes

It can be seen that if fountain codes are to be implemented in a RLNC environment, all the intermediate network nodes are not able to simply create random linear coded packets for transmission. BP decoding at the receivers require the encoding of packets to be according to the RS degree distribution. Thus an encoding method that prevents the occurrence of degree degeneration needs to be implemented at intermediate nodes allowing for the successful implementation of fountain codes in a RLNC environment.

A method called LT network codes (LTNC) was suggested in [8] where each intermediate network node is forced to encode packets according to the specified RS degree distribution. The implementation of LTNC allows for BP at the receiver nodes of the network, but the algorithm implemented at each intermediate node is of very high complexity as it runs sub-optimal coding and refining steps.

An improvement to LTNC, called Hybrid-LT network coding (H-LTNC), was proposed in [7]. In H-LTNC most of the intermediate nodes implement RLNC and only nodes connected to the receiver nodes implement a simplified LTNC encoding algorithm. It was found that the implementation of the LTNC algorithm at all the intermediate nodes is unnecessary as the receivers only obtain packets from network nodes they are connected to. The use of H-LTNC in a RLNC network reduces the encoding complexity at the intermediate network nodes and still allows for the implementation of BP decoding at the receiver nodes.

In this paper we present an improved H-LTNC, Enhanced H-LTNC (EH-LTNC) to ensure accurate encoding of packets of the needed target degrees at intermediate network nodes. This optimisation reduces the number of additional packets required for decoding and reduces the decoding delay of the method.

III. HYBRID-LT NETWORK CODES

In this section we shall provide a brief description of the H-LTNC method at intermediate nodes as presented in [7].

Consider n source packets of size m in a finite field $\mathbb{F}$ of size q. Each intermediate node $v \in \mathcal{V}$ collects packets from its incoming edges e', $\boldsymbol{y}(e'_u)$, where u is the number of incoming edges. As these packets arrive at the node, they are stored in a buffer. As soon as the node is presented with a transmission opportunity, an outgoing packet $\boldsymbol{y}(e)$ is created to be transmitted on the outgoing edges e. This outgoing packet is a linear combination of the packets present in the coding buffer of the node:

$$\boldsymbol{y}(e) = \sum_{e'} \boldsymbol{\beta_e}(e')\boldsymbol{y}(e') \quad (2)$$

where $\boldsymbol{\beta_e}$ is the local encoding vector of packet $\boldsymbol{y}(e)$.

The aim of this method is to employ RLNC as far as possible in the network, while still ultimately implementing a low complexity BP decoding algorithm. The BP decoding algorithm relies on the statistical properties of the encoded packets collected from the incoming edges at a network receiver node, which are in the form

$$\boldsymbol{y}(e') = \sum_{i=1}^{n} \boldsymbol{g}_{e'} \boldsymbol{x}_i \quad (3)$$

with $\boldsymbol{g}_{e'}$ the global encoding vector of received encoded packet $\boldsymbol{y}(e')$.

When the degrees of the received packets $d[\boldsymbol{y}(e'_u)]$ adhere to the RS distribution, BP can be implemented successfully.

In H-LTNC, before an encoded packet can be transmitted from one node to another, a connection is established between the neighbouring nodes. The receiving node transmits a message via the feedback channel stating whether it is a receiver node or not. Based on the feedback information, each node is categorised as a *random coding node* or *fountain coding node* and then proceeds with the suitable encoding algorithm.

A. Random coding nodes

When the connection established by an intermediate node is not with a receiver, the node implements low complexity RLNC for packet encoding of order $\mathcal{O}(mb)$. The local encoding vectors $\boldsymbol{\beta_e}$ as shown in (2) are chosen randomly and independently from $\mathbb{F}_q$ to construct an encoded packet $\boldsymbol{y}(e)$ of a random degree $d[\boldsymbol{y}(e)]$.

B. Fountain coding nodes

When the connection established by an intermediate node is with a receiver node, the encoding node applies a different encoding procedure so that the receiver node receives packets encoded according to the RS degree distribution. This method is formally presented in [7].

Firstly, the receiver node draws a target degree d_T from the RS distribution and communicates this value to the fountain coding node. The fountain coding node then examines the encoded packets $\boldsymbol{y}(e'_u)$ in its buffer and the degrees of the packets in the buffer are determined $d[\boldsymbol{y}(e'_u)]$.

If a packet of the target degree d_T is present in the buffer it is selected as the new outgoing packet where

$$\boldsymbol{y}(e) = \boldsymbol{y}(e'_i) \quad (4)$$

and $d[\boldsymbol{y}(e'_i)] = d_T$. Thus the node only acts as a forwarding node and runs an algorithm of order $\mathcal{O}(mb)$.

If there is no packet of d_T in the buffer of the node, packets whose linear combination can produce a packet where $d[\boldsymbol{y}(e)] = d_T$ are selected for encoding of $\boldsymbol{y}(e)$. When the target degree d_T cannot be reached, the packet with the closest degree to d_T is used. This encoding method is complex and scales exponentially.

This encoding algorithm employed at the fountain coding nodes enables the use of BP decoding at the receiver nodes due to the arriving packets being from the RS degree distribution. The standard method for BP decoding described in Section II *B* is implemented at the receiver nodes for decoding.

IV. Enhanced H-LTNC

In [7] it was shown that the H-LTNC method enables the successful use of BP decoding at the receiver nodes. The disadvantage of this method, however, was an increase in the additional packets needed to be collected by the receiver nodes before BP decoding could be completed successfully. This resulted in a longer decoding delay at the receiver nodes when compared to LTNC where all nodes are forced to encode packets according to the RS degree distribution.

The reason for the requirement of more additional packets and the longer decoding delay was a result of a received degree distribution that does not match that of the required RS distribution. The encoding method employed at the fountain coding nodes was not optimally constructed in order to produce the required distribution, resulting in sub optimal decoding. We now present two modifications to the intermediate network nodes to ensure the accurate encoding of packets of the needed target degrees. This optimisation reduces the number of additional packets required for decoding which in turn will render minimum decoding delays.

A. Sparse RLNC

The first improvement made to the H-LTNC method is to allow random coding nodes to employ sparse RLNC. Previously in the random coding nodes, packets were encoded randomly, but non-sparse. The probability of successful decoding for sparse RLNC is comparable to that of traditional RLNC when coding is done in a large finite field $\mathbb{F}_q$ and the density of non-zero symbols in the global encoding vectors $\boldsymbol{g}_{e'}$ are greater than a certain threshold value [11].

In [7] when the fountain coding nodes received non-sparse packets to encode a packet of a low degree, the target degree was frequently not attainable. This interfered with the statistical properties of the packets needed for BP decoding.

In our network scenario the fountain coding nodes are required to construct packets of mostly low degrees adhering to the RS degree distribution. As shown in (2), the local encoding vector $\boldsymbol{\beta_e}$ for each encoded packet $\boldsymbol{y}(e)$ formed is chosen from a sufficiently large finite field $\mathbb{F}_q$. As an improvement, the encoding vectors are chosen to be sparse so that the average degrees of the encoded packets remain low. Thus when fountain coding nodes receive encoded packets of relatively low degrees, the construction of a packet of a low degree from the RS degree distribution is simplified greatly and a packet of the target degree can be constructed successfully with high probability.

At the random coding nodes when RLNC are performed

with sparse linear combinations the encoding complexity at the nodes is also reduced.

B. Buffer flushing policy

In a wireless network environment the buffers of the intermediate nodes are flushed periodically according to a flushing policy [10]. Thus packets received at the incoming edges of a node are stored in the buffer and then flushed from it after a certain time has passed. This allows for the periodic construction of new encoded packets consisting of possibly new source packets.

In our network environment modelled by a random geometric graph (RGG) with R nodes and a minimum cut between source and receiver nodes of min-cut $(s, z) \geq n$, the average number of incoming edges per intermediate nodes are $|e'|_{ave} = \sqrt{R}$. In the previous work done in [7] the flushing policy of the network was set to flush the nodes' buffers at intervals equating to the reception of approximately $\sqrt{R}$ packets. Thus each node must construct a new encoded packet from approximately $\sqrt{R}$ received packets. For the fountain coding nodes that must construct a packet of a specific degree, the limited number of packets in its buffer can limit the success of packet encoding. Adjusting the flushing policy of these nodes to flush incoming packets at less frequent intervals, the buffers would contain more packets. This gives each fountain coding node a wider selection of packets which would enable it to construct a packet of a specific degree more accurately.

V. Simulation Results

In this section we evaluate the BP decoding performance when different encoding methods are implemented in the RLNC network environment. We evaluate the decoding delay, the received degree distribution and the encoding complexity for each encoding method used.

As discussed in Section IV *B*, we consider a network environment that can be modelled by a random geometric graph (RGG) with $R = 100$ nodes and a single source s and receiver node z. The minimum cut between source and receiver nodes is min-cut $(s, z) \geq n$. The data transmitted by the source to the receiver consists of approximately 10000 packets in the finite field $\mathbb{F}_{2^8}$. These packets are divided into n transmission packets $\{x_i\}_{i=1}^{n}$ of size m. We conducted 1000 Monte-Carlo simulations for various values of n. This experimental setup is based on that of [10].

We consider a multicast communication network and assume a feedback channel allowing communication between nodes regarding connectivity to receiver nodes. The receiver node implements low complexity BP decoding.

A. Decoding delay

Decoding delay can be seen as the elapsed time between the reception of a packet at a receiver node and the decoding thereof [12]. When packets are received that adhere to the RS distribution, the decoding delay should be equal to zero as this distribution ensures optimal decoding.

We denote t as the timestep of the simulation when z obtains a new packet from the network. We denote the global rank of the network as R_n, which is equal to the number of source packets n. The rank present at receiver node z at time t is defined as $R_z(t)$. The source packets decodable by node z are defined as *effective packets* and the total number of effective packets at z up to time t is denoted as $E_z(t)$.

Fig. 1 shows the normalised $E_z(t)/R_n$ decoding curves for H-LTNC, EH-LTNC and a simplified version of LTNC for $n = 35$.

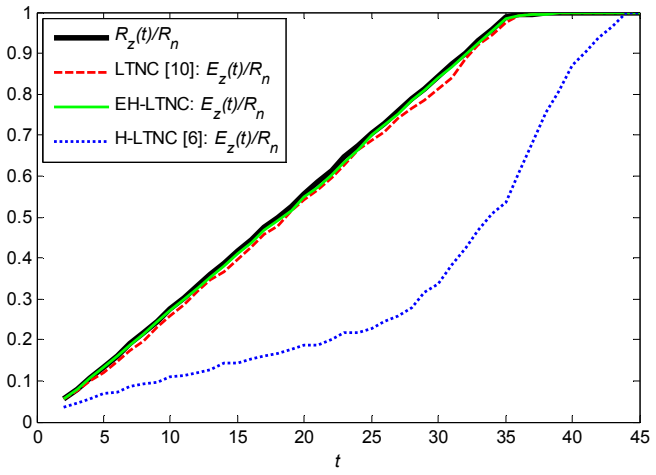


Figure 1: Decoding delay for BP decoding for $n = 35$

The curve $R_z(t)/R_n$ shows the normalised value of the rank available at z, which expresses the total number of source packets possibly decodable at time t. This curve gives the lower limit of decoding delay for any system at time t.

The graph shows that EH-LTNC renders a large improvement in the decoding delay compared to H-LTNC. Where the H-LTNC method has an approximate decoding delay of $t = 10$, the decoding delay of EH-LTNC is approximately zero. This shows that EH-LTNC is an accurate encoding method to produce packets suitable for BP decoding in a RLNC network. When compared to the LTNC method, the results are approximately the same.

B. Received degree distributions

Next we evaluate the degree distributions of the packets obtained by the receiver nodes for each encoding method. Fig. 2*a* shows the RS degree distribution for $n = 35$ where $c = 0.2$ and $\delta = 0.5$. The degree distribution of the received packets are shown in Fig. 2 *b,c,d* for the implementation of H-LTNC, EH-LTNC and a simplified version of LTNC.

It can be seen that H-LTNC produces a degree distribution

that is not comparable to the RS distribution. EH-LTNC produces packets with degrees that are comparable to the RS distribution, which shows that the improvement of sparse encoding and extended flushing times allow for the accurate encoding of packets from the RS distribution. This corresponds to the results shown in Fig. 1 that the production of packets of the RS distribution is done accurately and that packets are decoded successfully via BP decoding with minimal decoding delay.

The results of EH-LTNC are also comparable to the computationally complex LTNC method which supports the findings depicted in Fig. 1.

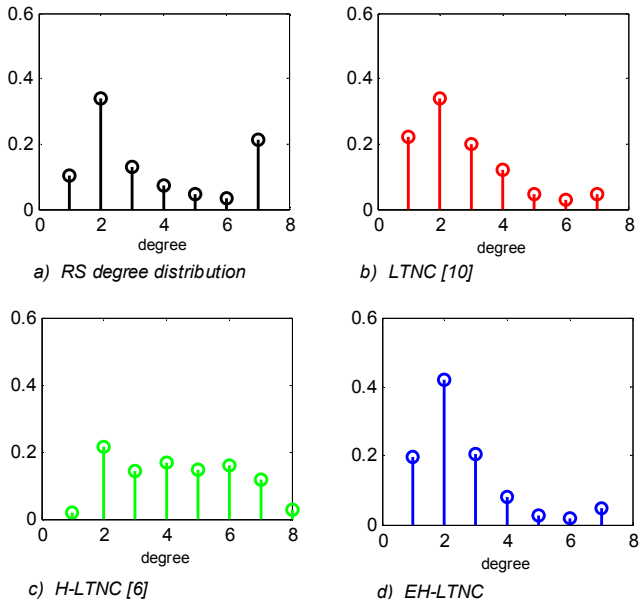


Figure 2: Received degree distributions

C. Encoding complexity

The EH-LTNC method produces the same results for decoding as the LTNC method. The advantage of the presented method, however, is that it has a lower encoding complexity than LTNC.

The LTNC method requires a complex encoding algorithm at all the intermediate network nodes. In our network environment with a min-cut $(s, z) \geq n$, the effective number of incoming edges of a receiver node are $|e'| = n$. Thus most of the intermediate nodes perform RLNC and only nodes connected to a receiver implement the complex encoding algorithm. With a network of R nodes and min-cut $(s, z) \geq n$, approximately n nodes are fountain coding nodes and $(R - n)$ are random coding nodes. Thus the relationship between network size R and n determines the encoding advantage of EH-LTNC over that of LTNC.

VI. Conclusion

In this paper we presented improvements to the H-LTNC method presented in [7]. The Enhanced H-LTNC method allows for the use of fountain codes in conjunction with RLNC in a communication network to allow low complexity BP decoding in a network that communicates at multicast capacity. The presented method allows low complexity linear encoding at most of the intermediate network nodes as well as a low decoding complexity with the use of BP decoding.

We showed that when RLNC are performed with sparse linear combinations and packet buffers are flushed at less frequent intervals, EH-LTNC renders a decoding delay which approaches that of the lower limit. This is because the method allows for the accurate encoding of packets that closely resembles the RS degree distribution enabling receiver nodes to successfully implement BP decoding. The EH-LTNC method retains the low complexity encoding of H-LTNC.

The presented method has the largest encoding complexity advantage in networks where the ratio between min-cut and number of nodes (n/R) is small. In wireless sensor networks information packets are traditionally small and may only consist of a few bits [13] where the data is transmitted to a sink via a group of intermediate nodes. Thus the wireless sensor network environment is suitable for the implementation of this method.

VII. Acknowledgement

This work was completed with funding from the Telkom Centre of Excellence at the North-West University, Potchefstroom Campus.

VIII. References

[1] M. Luby, “LT codes,” Foundations of Computer Science, in *Proc. The 43rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 271-280, 2002.

[2] A. Shokrollahi, “Raptor codes,” *IEEE/ACM Trans. Netw., IEEE Press*, Vol. 14, pp. 2551-2567, 2006

[3] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *IEEE Int. Symp. Information Theory*, Yokohama, pp. 442, July 2003.

[4] H. Wang, J. Goseling, J.H. Weber, "Network coded flooding, " Master’s thesis, Dept of Telecommunic., Delft University of Technology, June 2009.

[5] A. Hessler, T. Kakumaru, H. Perrey, D. Westhoff, “Data obfuscation with network coding,” *Computer Communications*, Nov 2010.

[6] S. Yang and R.W. Yeung, “Coding for a network coded fountain,” in *Proc. 2011 IEEE International Symposium on Information Theory (ISIT’11)*, Saint Petersburg, Russia, pp. 2583–2587, July 2011.

[7] S. von Solms and A.S.J. Helberg, “The implementation of LT network coding in resource limited RLNC networks,” in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*, East London, South Africa, 2011.

[8] M. Champel, K. Huguenin, A. Kermarrec, N. Le Scouarnec, “LT Network Codes”, *Institut National de Recherch en Informatique et en Automatique (INRIA)*, November 2009.

[9] R. Koetter and M. Medard, “Beyond routing: an algebraic approach to network coding,” in *Proc. Twenty-First Annual Joint Conf. of the IEEE Computer and Communications Societies*, vol. 1, pp. 122-130, 2002.

[10] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," *Allerton Conference on Communication, Control and Computing*, Monticello, IL, October 2003.

[11] M. Wang and B. Li, "How Practical is Network Coding?," *IEEE International Workshop on Quality of Service 2006*, Yale University, Connecticut, June 19–21, 2006.

[12] J. K. Sundararajan, D. Shah and M. Medard, "Online network coding for optimal throughput and delay – the three receiver case," in *Proc. Int. Symp. on Inform. Theory and its Applic.*, Aukland, New Zealand, 2008.

[13] M. Jafari, L. Keller, C. Fragouli and K. Argyraki, "Compressed Network Coding Vectors," in *Proc. IEEE International Symposium on Information Theory (ISIT 2009),* Seoul, Korea, pp. 109-113, June 2009.

RANDOM LINEAR NETWORK CODING FOR BELIEF PROPAGATION DECODING

S. von Solms [a b] and A.S.J. Helberg [a]

[a] School for Electric, Electronic and Computer Engineering, North-West University, Potchefstroom Campus, Potchefstroom, South Africa
[b] corresponding author

E-mail: sune.vonsolms@nwu.ac.za, albert.helberg@nwu.ac.za
Fax: +27(0)18 299 1977, Phone: +27(0) 18 299 1978
Postal Address: Building N1-A, North-West University, Potchefstroom Campus, Hoffmann street, Potchefstroom

Abstract: Implementing Belief Propagation decoding methods at receiver nodes in Random Linear Network Coding networks rather than traditional matrix inversion has the advantage of smaller decoding delay and complexity. The penalty is an increase in network resource usage due to complex encoding methods at network nodes. We introduce and study an encoding method which allows the implementation of Belief Propagation decoding in a Random Linear Network Coding environment that improves on the existing encoding method in terms of network resource usage at network nodes.

Keywords: Belief propagation decoding, Luby Transform codes, Network Coding, Random Linear Network Coding.

1. INTRODUCTION

In a network environment one challenge is to provide effective communication between source and receiver nodes so that the maximum capacity can be achieved and unreliable communication channels can be overcome. A method to utilize the capacity in decentralised networks is called Random Linear Network Coding (RLNC) [1]. Implementation of RLNC yields practical solutions for decentralised networks with limited resources [2, 3] and allows improved network utilisation [4]. Fountain codes which include Luby Transform (LT) codes [5] and Raptor codes [6] are ideal for erasure channels where no feedback is needed from the receiver nodes to the source regarding retransmission of erased packets.

For LT coding the source transmits encoded packets consisting of linear combinations of a subset of the n source packets with the degree of each packet determined by the Robust Soliton (RS) distribution. These packets are continually transmitted via intermediate network nodes until all the receivers have successfully decoded the source information through Belief Propagation (BP) [5].

A RLNC environment, however, relies on the random encoding of packets at the intermediate network nodes. Source packets of the RS distribution cannot retain their distribution characteristics when transmitted over a network with RLNC. The random linear encoding at the intermediate network nodes leads to degree degeneration where the specified input degree distribution degenerates with each random recoding at intermediate nodes in the network [7] so that the BP decoding at the receivers fail [8].

The authors in [9] developed an encoding procedure, called LT network codes (LTNC) for content dissemination networks. This method forces all intermediate network nodes in a decentralised network to encode their packets according to the RS distribution where possible, so that no degree distribution degeneration would appear. This method allows low complexity BP at the receiver nodes but encoding proves to be resource intensive. Another method was introduced in [8] to transmit data over a communication network where the receivers also implement BP decoding. These codes require encoding steps of linear complexity but a pre-decoding step with quadratic complexity.

In a network environment where data is only communicated between source and receiver, the LTNC method can be adapted and improved to allow linear encoding at most of the intermediate network nodes and low complexity BP decoding at the receivers. In this paper, we propose a hybrid encoding method that reduces the use of encoding resources at intermediate network nodes while maintaining the low complexity BP decoding scheme.

2. BACKGROUND

2.1 Random Linear Network Coding

Consider an acyclic network $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$ consisting of a single source node $s \in \mathcal{V}$ and a set of sink nodes $Z = \{z_1, \dots, z_{|z|}\}$, $Z \subset \mathcal{V}$. The source data is divided into n packets, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n]$ where $\boldsymbol{x}_i$ represents the ith source packet of size m in a finite field $\mathbb{F}$ of size q. These source packets are multicast over the edges $\mathcal{E}$ of the network from s. The achievable rate at which s can multicast the source packets reliably to the set of receivers Z is $r(s, Z)$. The maximum flow of the network for any $z \in Z$ is the upper bound on $r(s, z)$, thus min-cut $(s, z) \geq n$ [10].

Each intermediate node $v \in \mathcal{V}$ collects packets from its incoming edges e' and stores it in a buffer. As soon as the node is presented with a transmission opportunity, an outgoing packet is created to be transmitted on the outgoing edges e. The outgoing packet is created by randomly and linearly combining, through X-OR operations, the packets stored in the node buffer. The encoding complexity at each intermediate node equals $\mathcal{O}(mb)$, where b is the size of the buffer [8]. In the header of each transmitted packet is a coding vector of length n, describing the included source packets $\boldsymbol{x} \subseteq \boldsymbol{X}$ [11].

This action is repeated until the packets reach the receiver nodes. Each receiver node $z \in Z$ collects a set of $N \geq n$ encoded packets $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_N]$ from the network. These received packets has the form

$$\boldsymbol{y}_j = \sum_{i=1}^{n} g_{ij} x_i \ , j = 1,2, \dots, N \tag{1}$$

where $\{g_{ij}\}$ are the random coefficients from $\mathbb{F}_q$ forming a vector of size n, describing the source packets included $\boldsymbol{x} \subseteq \boldsymbol{X}$ [11]. This vector $\boldsymbol{g}_j$ forms a column vector of a $n \times N$ matrix $\boldsymbol{G}$ where

$$\boldsymbol{X} \times \boldsymbol{G} = \boldsymbol{Y}. \tag{2}$$

The finite field $\mathbb{F}_q$ of the global encoding vectors is sufficiently large so that $\boldsymbol{G}$ is invertible with high probability when N is only slightly larger than n. The solution of the linear system of equations in (2) decodes the source packets $\boldsymbol{X}$ [11].

Traditionally the decoding method employed in RLNC networks is Gaussian Elimination (GE) whose matrix inversion algorithm is computationally complex and of order $\mathcal{O}(n^3 + mn^2)$ [8]. This high complexity decoding scheme is not ideal to implement at receiver nodes as it is both time and resource consuming. Thus the need exist for a less complex decoding method in this environment.

2.2 Belief Propagation Decoding

Belief Propagation decoding is a low complexity decoding algorithm of order $\mathcal{O}(mnlogn)$ and is dependent on the statistical properties of the encoded packets. All packets must adhere to the RS degree distribution. A degree distribution specifies the number of source packets that must be included in each packet. For example: the degree of the encoded packet $y_i = \{x_1, x_2, x_4\}$ is $d_{y_i} = 3$.

This degree distribution ensures that decoding can commence as soon as a single-degree packet is received. The decoding process can be described by the following algorithm [5]:
Find an encoded packet, $y_j, 1 \leq j \leq N$, that only contains a single source packet, $x_i, 1 \leq i \leq n$.
Set source packet $x_i = y_j$ and delete y_j.
Subtract the value of x_i from all the other encoded packets $\{y_p\}_{p=1,p\neq j}^{N}$ that contains source packet x_i.
Repeat from (1) until all source packets $x_i, i \in \{1, n\}$ are determined.

It can be seen that the specific degree distribution is imperative to the success of the decoding method. A decentralised network cannot retain this distribution when RLNC is implemented al all the network nodes.

2.3 LT network coding [9]

LTNC is a method developed to combat degree degeneration in a decentralised network environment. In order to do so, all intermediate network nodes must implement a complex encoding algorithm ensuring that all transmitted packets adhere to the RS distribution. The encoding steps of LT network codes consist of two steps: *recoding* and *refining*. The encoding method is described briefly.

2.3.1 Recoding

An intermediate node draws a target degree d_T from the RS distribution and examines the degrees of the encoded packets in its buffer. The target degree may not be reachable, as the maximum reachable d_T is upper bounded by the number of source symbols present in the node buffer. If the value of d_T is higher than the number of source packets present in the buffer, a new target degree is selected.

The recoding of a new d_T-degree packet involves a sub-optimal solution where packets are examined according to decreasing orders of degrees. A packet with a degree closest to d_T is selected and packets are X-OR'ed when the addition of the specific packet increases the degree but keeps it smaller or equal than d_T. This process iterates until the target degree is reached. When the target degree cannot be reached, the encoded packet with the degree closest to d_T is selected.

2.3.2 Refining

After the recoding step, the new encoded packet is refined by the replacement of frequent source packets with less frequent packets. This is done in order to reduce the variance of the degree distribution of the native packets used. The information regarding the frequency of the transmitted source packets are available in a network wide data structure which is updated each time a new

encoded packet is transmitted. This method iteratively replaces frequently occurring source packets with less frequent ones by using decoded source packets or encoded packets of degree 2 stored in a buffer of each intermediate node.

2.3.3 Disadvantages of LTNC

It can be seen that the algorithm implemented at each intermediate node has high complexity as it runs sub-optimal coding and refining steps. The refining step requires that all intermediate nodes need to keep 1 and 2 degree packets in a buffer for the replacement of source packets in the refining step. This requires that all the buffers of the intermediate nodes are to be larger than in a traditional RLNC environment. The refining step also requires a centralised system to record the frequency of the source packets transmitted, which is not practical in a decentralised RLNC network.

In this paper, we reduce the resource usage in the network by only applying the recoding step of the LTNC algorithm in to intermediate nodes directly connected to receiver nodes. The other network nodes use RLNC. This hybrid-LTNC method decreases the computational requirements of the network nodes and eliminates the need for a centralised system.

3. HYBRID- LTNC

From Section 2 we find that the Belief Propagation decoding algorithm only relies on the statistical properties of the encoded packets collected at the network receiver nodes. Thus the implementation of the LTNC algorithm at all the intermediate nodes is unnecessary as the receivers only obtain packets from network nodes they are connected to. We can encode the data randomly and only enforce the encoding of RS distributed packets once they are required: a single hop from the receiver nodes.

In this paper we introduce an improvement to LTNC, called Hybrid-LT network coding (H-LTNC). The Hybrid-LT network coding method reduces the resource usage in the network, but still allows the implementation of BP decoding through the delivery of packets that statistically match the RS degree distribution at the receiver nodes. To achieve this, the recoding step of LTNC algorithm is only implemented at intermediate nodes a single hop from the receiver nodes in order to allow the reception of packets from the RS degree distribution. The other network nodes use RLNC to encode their packets.

In this situation where only the nodes closest to the receivers implement LTNC, we regard the implementation of the refining step of LTNC redundant. If the network is well connected, we accept that RLNC encoding will ensure that the packets arriving at the last-hop nodes will have an acceptable variance. The elimination of this step not only simplifies the encoding method which decreases the computational requirements, but eliminates the need for a network wide structure as well as additional storage buffers at the intermediate nodes.

In H-LTNC, before an encoded packet can be transmitted from one node to another, a connection is established between the neighbouring nodes. The receiving node transmits a message via the feedback channel stating whether it is a receiver node or not. Based on the feedback information, each node is categorised as a *RLNC node* or *LTNC node* and then proceeds with the suitable encoding algorithm. The process applied at each intermediate network node is discussed subsequently.

3.1 RLNC nodes

When the connection established by an intermediate node is not with a receiver, the node implements low complexity RLNC for packet encoding. The local encoding vectors are chosen

randomly and independently from $\mathbb{F}_q$ to construct an encoded packet of a random degree. This method is discussed in Section 2, has been thoroughly presented by many researchers [1, 2, 11].

3.2 LTNC nodes

When the connection established by an intermediate node is with a receiver node, the following encoding procedure is followed to ensure that the receiver node receives packets according to the RS degree distribution. This method is based on the recoding step of LTNC and the steps are discussed subsequently.

Firstly, the receiver node draws a target degree d_T from the RS distribution and communicates this value to LTNC node. The LTNC node then examines the degrees of the packets in its buffer. If a packet of the target degree d_T is present in the buffer it is selected as the new outgoing packet. If no packet of target degree d_T is found, LTNC nodes implement the sub-optimal iterative process to construct a new packet of degree d_T. This step builds a new encoded packet of degree d_T from a set of received packets so that the sum of their degrees equals d_T. The difficulty of this problem lies in the fact that the sum of the packets' degrees does not necessarily equal the degree of the sum of the packets. For example: a packet encoded from packets (x_1, x_2, x_4) and (x_3, x_4), renders a packet of degree 3 and not 5.

The packets are evaluated in decreasing order of degrees, starting from d_T, where a packet is selected and named, t. Iteratively, packets are added to t, evaluating the resulting degree, d_t. When the addition of a packet increases d_t where $d_t \leq d_T$, the new packet is added to t. This process repeats until the target degree d_T is reached. When the target degree cannot be reached, the packet, t, with the closest degree to d_T is used.

3.3 Reachability of RS distribution at receivers

Evaluating the degree distribution of the packets received at the receiver nodes for H-LTNC and LTNC, it was found that the required RS distribution was reached more successfully when all the network nodes implement LTNC, instead of only those connected to the receiver nodes (H-LTNC). This results in suboptimal BP decoding for H-LTNC as more packets are required at the receivers for decoding. The need for additional encoded packets at the receiver nodes lead to an increase in decoding delay. The decoding delay attained by the method can be seen in Fig. 1 in Section 4.

We determined that the encoding method employed at the LTNC nodes in H-LTNC was not optimally constructed in order to produce packets that match the RS distribution. In this situation the LTNC nodes must construct packets of mostly low degrees (RS distribution) from packets with normally distributed degrees received from the RLNC nodes. Encoding a low degree packet from packets with higher degrees frequently makes the target degree not attainable that interferes with the statistical properties of the packets needed for BP decoding.

We now present two modifications to the intermediate network nodes to improve the accurate encoding of packets at LTNC nodes. This optimisation reduces the number of additional packets required for decoding which in turn will render minimum decoding delays.

3.3.1 Sparse RLNC

The first improvement made to the H-LTNC method is the employment of sparse RLNC encoding at the RLNC nodes. In [12] it was shown that the efficiency of sparse RLNC is comparable to that of traditional RLNC when coding is done in a large finite field $\mathbb{F}_q$ and the density of non-zero symbols in the global encoding vectors $\boldsymbol{g}_j, j = 1,2, \dots, N$ are greater than a certain threshold value.

The encoding vectors at RLNC nodes are chosen to be sparse so that the average degrees of the encoded packets remain low. Thus when LTNC nodes receive encoded packets of relatively low degrees, the construction of a packet of a low degree from the RS degree distribution is simplified and a packet of the target degree can be constructed successfully with high probability. Also, when sparse RLNC is employed, the encoding complexity at the RLNC nodes is reduced.

3.3.2 Buffer flushing policy

In a wireless network environment the buffers of the intermediate nodes are flushed periodically according to a flushing policy [11]. Thus packets received at the incoming edges of a node are stored in the buffer and then flushed after a certain time has passed. This allows for the periodic construction of new encoded packets consisting of possibly new source packets.

For the LTNC nodes that must construct a packet of a specific degree, the limited number of packets in its buffer can limit the success of packet encoding. Adjusting the flushing policy of these nodes to flush incoming packets at less frequent intervals, the buffers would contain more packets. This gives each LTNC node a wider selection of packets which would enable it to construct a packet of a specific degree more accurately.

4. EVALUATION

Next, we evaluate the performance by comparing the proposed H-LTNC to RLNC and LTNC in a decentralised network.

4.1 Experimental setup

We consider a network environment that can be modelled by a random geometric graph (RGG) with $R = 100$ nodes and a single source s and receiver nodes $Z \subset \mathcal{V}$. The minimum cut between source and receiver nodes is min-cut $(s, z) \geq n, z \in Z$. The data transmitted by the source to a receiver consists of approximately 10000 packets in the finite field $\mathbb{F}_{2^8}$ where packets are divided into n transmission packets $\{x_i\}_{i=1}^{n}$ of size m and coding is performed over $\mathbb{F}_2$. We conducted 1000 Monte-Carlo simulations for various values of n, varying the size of m accordingly. This experimental setup is based on that of [3, 11].

We consider a multicast communication network and we assume a feedback channel allowing communication between nodes regarding connectivity to receiver nodes.

4.2 Decoding delay

Decoding delay can be seen as the elapsed time between the reception of a packet at a receiver node and the decoding thereof [13]. When packets are received that adhere to the RS distribution, the decoding delay should be equal to zero as this distribution ensures optimal decoding.

We denote t as the timestep of the simulation when a receiver z obtains a new packet from the network. We denote the global rank of the network as R_n, which is equal to the number of source packets n. The rank present at receiver node z at time t is defined as $R_z(t)$. The source packets decodable by node z are defined as *effective packets* and the total number of effective packets at z up to time t is denoted as $E_z(t)$.

Fig. 1 shows the normalised $E_z(t)/R_n$ decoding curves for the improved and unimproved H-LTNC and LTNC for $n = 35$ at a randomly selected receiver node $z \in Z$. The curve $R_z(t)/R_n$ shows the

normalised value of the rank available at z, which expresses the total number of source packets possibly decodable at time t. This curve gives the lower limit of decoding delay for any system at time t. The curves of LTNC, H-LTNC and die unimproved H-LTNC are all curves showing the number of effective packets at time t, $E_z(t)/R_n$.

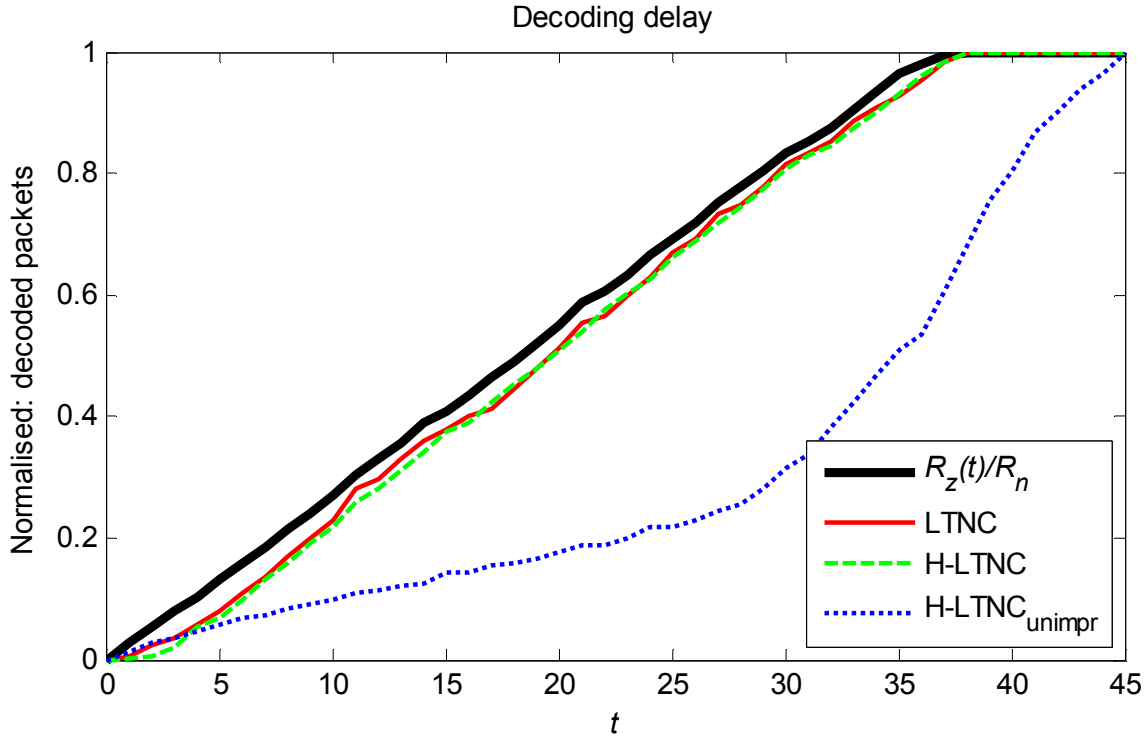


Figure 1: Decoding delay for BP decoding for $n = 35$

The first curve, $R_z(t)/R_n$, shows the normalised value of the rank available at the receiver node. This is the upper bound of the number of effective packets possible at each timestep. Note that the rank of the $R_z(t)/R_n$ curve is not 35 at timestep 35 as approximately 2 non-innovative packets are received during the reception of the encoded packets. As encoding takes place over $\mathbb{F}_2$, this is justified [15].

The graph shows that the LTNC method has a small decoding delay, where all the packets are decoded through BP when sufficient packets are received, $R_z(t)/R_n = 1$. This supports the findings in [6] that the LTNC encoding process delivers packets of RS distribution to the receiver nodes.

When evaluating the decoding delay curve of H-LTNC, it can be seen that these exist a small decoding delay, but all packets are decoded when $R_z(t)/R_n = 1$.It can be seen that sparse RLNC and better buffer flush times sustains the accurate encoding of RS distributed packets for H-LTNC.

4.3 Encoding complexity

From the results it can be seen that employing the LTNC encoding procedure only at strategic network nodes results in the same distribution of packets as employing the complex LTNC method at all the network nodes. The advantage of the presented method, however, is that it has a lower encoding complexity than LTNC.

As described in Section 2, the encoding complexity of RLNC at each intermediate node equals

$$\mathcal{O}(mb) \tag{3}$$

where b is the size of the buffer [8].

The recoding step of LTNC, however, is suboptimal as all the packets in the buffer are compared to each other until a packet of the target degree can be constructed. The comparison of b packets to each other can require a maximum of

$$m(b-1) + m(b-2) \ldots + m = \frac{mb(b-1)}{2} \tag{4}$$

arithmetic operations. From the abovementioned formula it can be seen that the complexity of the recoding step of LTNC is $\mathcal{O}(mb^2)$. The complexity of the refining step of LTNC is not influential as it is of smaller order than the recoding step.

The LTNC method implements the recoding step at all the intermediate network nodes, thus the total encoding procedure of LTNC is of order:

$$\mathcal{O}(Rmb^2) \tag{5}$$

where R is the number of nodes in the network.

The encoding complexity of H-LTNC can be computed through the combination of (3) and (4). In our network environment with a min-cut $(s, z) \geq n$, the number of incoming edges at each receiver node are $|e'| = n$. With $|Z|$ receivers in the network, the minimum and maximum number of possible last-hop nodes in the network is n and $n|Z|$, respectively.

Thus the complexity of the encoding procedure of EH-LTNC can be approximated by

$$\begin{gathered}\mathcal{O}(nmb^2) + \mathcal{O}((R-n)mb) \\ = \mathcal{O}(nmb^2)\end{gathered} \tag{6}$$

for $(R-n)$ intermediate nodes being RLNC nodes, and

$$\begin{gathered}\mathcal{O}(|Z|nmb^2) + \mathcal{O}((R-|Z|n)mb) \\ = \mathcal{O}(|Z|nmb^2).\end{gathered} \tag{7}$$

for maximum number of $(R-|Z|n)$ intermediate nodes being RLNC nodes.

It can be seen that the relationship between the number of receiver nodes and the total number of last hop nodes determines the encoding advantage of H-LTNC over that of LTNC.

5. H-LTNC WITH PRECODING

H-LTNC method can be further improved by using the same technique as Raptor codes where the source packets are precoded using a traditional erasure code. The belief propagation decoding of H-LTNC now only requires the decoding of a constant fraction of the transmitted packets, where the erasure code enables the receiver to recover the original source information in the presence of possible packet erasure.

The requirement of decoding only a constant fraction of the transmitted packets reduces the decoding complexity at the receiver nodes from $\mathcal{O}(mnlogn)$ to a linear complexity of $\mathcal{O}(mn)$ [8].

6. CONCLUSION

In this paper we presented a coding method constructed through the combination random linear network coding and Luby Transform codes for the implementation of Belief Propagation decoding in networks that implement RLNC.

The Hybrid-LTNC method is based on LTNC and the implementation of H-LTNC in a RLNC network reduces the computational complexity of the network at the intermediate network nodes and allows for the implementation of a low complexity Belief Propagation at the receiver nodes.

Through Monte Carlo simulations it is determined that the H-LTNC method shows an improved performance over LTNC in terms of computational resources at the intermediate network nodes. The presented method has the largest encoding complexity advantage in networks where the ratio between min-cut and number of nodes (n/R) is small. In wireless sensor networks information packets are traditionally small and may only consist of a few bits [14] where the data is transmitted to a sink via a group of intermediate nodes. Thus the wireless sensor network environment is suitable for the implementation of this method.

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 IEEE Int. Symp. Information Theory, Yokohama, July 2003, p. 442.

[2] D. Silva, F. R. Kschischang, and R. Koetter, "A Rank-Metric approach to error control in random network coding," Proc. Information Theory for Wireless Networks, 2007 IEEE Information Theory.

[3] T. Ho, "Networking from a Network Coding perspective," PhD Thesis, Massachusetts Institute of Technology, Dept of EECS, May 2004.

[4] H. Wang, J. Goseling, J.H. Weber, "Network coded flooding, " Master's thesis, Dept of Telecommunic., Delft University of Technology, June 2009.

[5] M. Luby, "LT codes," Foundations of Computer Science, in Proc. The 43rd Annual IEEE Symposium on Foundations of Computer Science, pp. 271-280, 2002.

[6] A. Shokrollahi, "Raptor codes," IEEE/ACM Trans. Netw., IEEE Press, Vol. 14, pp. 2551-2567, 2006.

[7] A. Hessler, T. Kakumaru, H. Perrey, D. Westhoff, "Data obfuscation with network coding," Computer Communications, Nov 2010.

[8] S. Yang and R.W. Yeung, "Coding for a network coded fountain," in Proc. 2011 IEEE International Symposium on Information Theory (ISIT'11), Saint Petersburg, Russia, pp. 2583–2587, July 2011.

[9] M. Champel, K. Huguenin, A. Kermarrec, N. Le Scouarnec, "LT Network Codes", Institut National de Recherch en Informatique et en Automatique (INRIA), November 2009.

[10] R. Koetter and M. Medard, "Beyond routing: an algebraic approach to network coding," in Proc. Twenty-First Annual Joint Conf. of the IEEE Computer and Communications Societies, vol. 1, pp. 122-130, 2002.

[11] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," Allerton Conference on Communication, Control and Computing, Monticello, IL, October 2003.

[12] M. Wang and B. Li, "How Practical is Network Coding?," IEEE International Workshop on Quality of Service 2006, Yale University, Connecticut, June 19–21, 2006.

[13] J. K. Sundararajan, D. Shah and M. Medard, "Online network coding for optimal throughput and delay – the three receiver case," in Proc. Int. Symp. on Inform. Theory and its Applic., Aukland, New Zealand, 2008.

[14] M. Jafari, L. Keller, C. Fragouli and K. Argyraki, "Compressed Network Coding Vectors," in Proc. IEEE International Symposium on Information Theory (ISIT 2009), Seoul, Korea, pp. 109-113, June 2009.

[15] O. Trullols-Cruces, J. M. Barcelo-Ordinas and M. Fiore, "Exact Decoding Probability Under Random Linear Network Coding," *IEEE Communications Letters,* 2011, 15, 67-69

Modified Earliest Decoding for Random Network Codes

Suné von Solms, Albert S.J Helberg
School for Electric, Electronic and Computer Engineering,
North-West University, Potchefstroom Campus, Potchefstroom, South Africa
E-mail: sune.vonsolms@nwu.ac.za, albert.helberg@nwu.ac.za

Abstract— **We present a practical, modified version of earliest decoding for random linear network coding networks. This decoding method has a lower complexity and decoding delay than traditional Gaussian elimination decoding schemes.**

Keywords- Earliest Decoding; Network Coding; Random Linear Network Coding.

I. INTRODUCTION

The study of network coding and the advantages they offer to various areas of communications, both in wired and wireless networks, has been extensively studied in the past years. The network coding ability of networks to recode received packets at intermediate nodes improves network performance through improvement in network throughput as well as energy efficiency and delay [2].

The problem, however, is the bridging of the gap between the theoretical work and the advantages they propose; and the practical implementation with the challenges they present.

The decentralized design of network coding, namely random linear network coding (RLNC) [1] allows for a more practical approach to network coding, where the need for centralized network control and planning is unnecessary. Random linear network coding allows random and independent recoding of receiver packets at intermediate network nodes. This allows the receiver node to decode the transmitted data upon reception of any set of random encoded packets of sufficient rank.

These decoding algorithms are of high complexity, time consuming and also subject to decoding delays. The effectiveness of random linear network coding in a practical setting is therefore dependent on the complexity and decoding delay of the implemented decoding algorithm [3, 12].

II. RELATED WORK

Random linear network coding relies on the ability of the receiver to receive a set of linear independently coded packets and their associated coding vectors in order to decode the transmitted packets. The use of Gaussian elimination (GE) for decoding k random linear network coding packets of size m at the receiver requires $\mathcal{O}(m \cdot k^2)$ operations [3]. When the number of transmitted packets increase, the generator matrix G, becomes larger and more complex. These factors cause an exponential growth in the decoding complexity. Gaussian elimination has a decoding delay equal to the length of time needed by the receiver to collect all sufficient packets. Thus decoding can only start once the receiver has collected a set of linearly independent packets proportional to the size of the transmission [3, 11]. This leads to a very high decoding delay in the case of large transmission sets.

A method proposed to decrease the decoding delay, is earliest decoding (ED) [2, 3, 11]. This process entails that Gaussian elimination is performed as soon as linearly independent packets are collected by the receiver node with sufficient rank. In contrast to traditional Gaussian elimination where the receiver has to wait for a full rank decoding matrix before decoding can commence, earliest decoding can be performed as soon as sufficient information exists in order to decode the first set of packets, even though the decoding matrix is incomplete. An example of this method is illustrated in Fig. 1.

It is clear that earliest decoding yields a smaller decoding delay. Also, the inherent divide and conquer approach of earliest decoding, leads to a faster decoding time, although the decoding complexity of earliest decoding still remains $\mathcal{O}(m \cdot k^2)$.

A method to reduce the use of computational resources at the receiver nodes is the use of Luby Transform (LT) codes [7]. LT codes allow for the use of a low complexity decoding scheme based on belief propagation in $\mathcal{O}(m \cdot k \cdot \log k)$ operations, which we refer to as avalanche decoding.

For this belief propagation decoding scheme, encoded symbols cannot be generated randomly, but according to a specified degree distribution, $\rho(d)$. The encoding algorithm

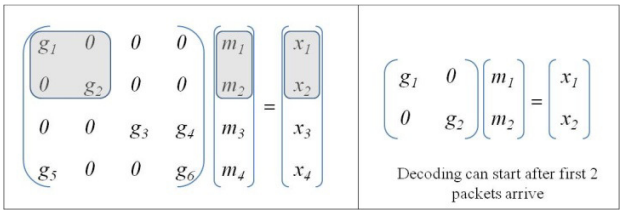


Figure 1. Example of earliest decoding [2]

guarantees efficient decoding times through the use of a low complexity decoding algorithm named avalanche decoding [7, 8]. This low complexity decoding scheme is not easily implemented in a decentralized random linear network coding environment, because encoding must take place according to a specific degree distribution. When implemented in a random linear network coding environment, the source packets cannot be encoded according to $\rho(d)$ and then retain their distribution characteristics when transmitted over a network with random linear network coding. Due to the random encoding properties of the network, so called degree distribution degeneration occurs, where the specified input degree distribution degenerates with each random recoding at intermediate nodes in the network [14].

In [10] the authors propose a method called LT Network Coding, where the intermediate network nodes encode the packets according to the predetermined distribution [7]. Each intermediate node evaluates and encodes each packet so that its degree distribution is maintained such that avalanche decoding can be implemented at the receiver node. This method leads to a significant reduction in decoding complexity compared to traditional Gaussian elimination.

The disadvantage of the LT Network Coding scheme is the increase of the overall communication overhead as well as the use of computational resources at the intermediate network nodes to enforce the needed degree distribution [7, 10]. In practical networks where the topology is unknown or constantly changing, LT Network Coding can become difficult to implement.

Also, in a system where possible packet erasures may occur, it is possible that Gaussian elimination will not be feasible due to the loss of sufficient linearly independent equations at the receiver nodes. Our proposed scheme reduces the impact of such erasures since limited decoding of packets from such generation is still possible.

III. MODIFIED EARLIEST DECODING

In contrast to enforcing the encoding of packets from $\rho(d)$, we introduce the concept of modified earliest decoding (MED) to approximate avalanche decoding of received coding vectors. This decoding scheme has a low complexity and a low delay by combining aspects of the low complexity of avalanche decoding and low decoding delay concept of earliest decoding. In contrast to avalanche decoding, this method does not rely on the decoder receiving a specific degree distribution based on the Hamming weight, d_w, of the coding vectors, but rather on a specific distribution of the Hamming distances, d_H, of the received vectors.

We show that this method can be implemented in a random linear network coding network without the need to encode or recode the packets according to a specific distribution. The following definitions are required:

Definition 1. The *Hamming distance* between two vectors $x, y \in \mathbb{F}_2^n$ is the number of coordinates that they differ and is denoted by $d_H(x, y)$, [13]

$$d_H(x, y) = \sum_{i=1}^{n} |x_i - y_i|. \quad (1)$$

Definition 2. The *Hamming weight* of a vector $x \in \mathbb{F}_2^n$ is the number of non-zero components, and is denoted by $d_w(x)$ [13].

A. Encoding

Consider a network represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The network consists of source node $S \in \mathcal{V}$ and a set of r receiver nodes $T = \{t_1, \dots, t_r\}, T \in \mathcal{V}$. The set of edges $\mathcal{E}$ represents the communication channels. An edge j carries the random encoded vector $Y(l)$ [1, 5, 6].

A source node, S, contains discrete k random independent packets $\{X_i\}_{i=1}^{k}$ from the finite field $\mathbb{F}_q$. These source packets are transmitted in a generation over a synchronized random linear network coding network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The intermediate network nodes generate packets $\{Y_j\}_{j=1}^{l}$ that are linear random combinations of the packets X_i received by the incoming edges of the node, (in contrast to LT Network Coding where the intermediate enforce a specified degree distribution on the linear combinations)

$$Y_j = \sum_{i=1}^{k} g_{ij} X_i \ , j = 1,2, \dots, l \quad (2)$$

where g_{ij} are the randomly generated coefficients from a finite field $\mathbb{F}_2$. The set of coefficients $g_{i1}, g_{i2}, \dots, g_{il}$ can be referred to as the *encoding vector* for Y_i and this vector is sent as a packet header in order to record the linear combination of the messages present in the received packet [3, 4].

The encoded packets collected by a receiver node, $\{t_d\}_{i=1}^{r} \in T$, are random linear combinations of the packets encoded in the network, represented as

$$Z(t_d, i) = \sum_{\{l: head(l) = t_d\}} g_{t_d, l} Y(l). \quad (3)$$

B. Description of decoding method

In traditional avalanche decoding the Hamming weights of the received encoding vectors are evaluated. When the Hamming weight is $d_w = 1$, it contains only a single source packet and can therefore be decoded by performing a linear combination with other coding vectors that also contain the specific packet. The remaining vectors are examined to find a new vector of $d_w = 1$ that can be used in the following iteration of the decoding algorithm.

The possibility, however, of obtaining a packet vector with $d_w(x) = 1$ where $x \in g_{t_d,l}$ is small when packets are randomly encoded. This makes the avalanche decoding method inefficient in a network with random linear network coding.

Consider two vectors $x, y \in g_{t_d,l}$. As we will show, the probability of these vectors having a Hamming distance of $d_H(x, y) = 1$, is significantly higher than either having weight $d_w = 1$. When these two packets are linearly combined, a resulting coding vector with $d_w = 1$ is obtained, which is then used for decoding in a similar way as with traditional avalanche decoding.

C. Decoding algorithm

The proposed decoding method is done iteratively through the use of the global encoding vectors $g_{t_d,l}$ of the received encoded packets $Z(t_d, i)$ at receivers $\{t_d\}_{i=1}^{r} \in T$.

The modified earliest decoding process operates in the following steps for every cycle:

1) Collect l new encoding vectors from network and store in decoding buffer.
2) Evaluate the Hamming distances, $d_H(x, y)$ of the $k \times l$ encoding vectors $g_{t_d,l}$, where $x, y \in g_{t_d,l}$:
   a) if no $d_H(x, y) = 1$ can be found: repeat from (1).
   b) if $d_H(x, y) = 0$, discard vector x or y.
   c) if $d_H(x, y) = 1$, decoding process can commence.
3) Calculate $z = x \oplus y$. The Hamming weights of x and y are determined. If $d_w(x) < d_w(y)$, then x is reinserted into the decoding buffer and z replaces y in the decoding buffer.
4) Repeat step (2) and (3) as long as a $d_H(x, y) = 1$ can be found, where $x, y \in g_{t_d,l}$.
5) Repeat from (1) until the identity matrix is present in decoding buffer.

This process is shown in Example 1. It should be noted that the avalanche decoding algorithm is a subset of our modified earliest decoding algorithm.

Example 1: The decoding process is illustrated in Fig. 2. The encoded packets collected by the receiver as well as their global encoding vectors are shown in (*I*). Through evaluation in (*II*) it is seen that $d_H(a, b) = 1$, and that $d_w(b) < d_w(a)$. Vector a is replaced by $a' = (a \oplus b)$, and vector b is returned to the decoding buffer (*III*). In the second iteration, vectors c and d are used, where c is replaced by $c' = (c \oplus d)$ and d returned. In (*IV*) vectors a' and d are used, and in the last iteration (*V*), vectors b and d'. It can be seen that the end result is a matrix with coding vectors of Hamming weight $d_w = 1$. This matrix can be rearranged to form an identity matrix.

Note: In this example, all the source symbols are transmitted in a single transmission, which does not require more than a single iteration of the method.

When compared to earliest decoding, it can be seen that the proposed method is a modified version of it. Modified earliest decoding is basically the forming of a sub-matrix from the global encoding matrix, just as earliest decoding, but ignoring the presence of source packets that are duplicated. Again consider Fig. 2 frame (*II*): earliest decoding would not have been used in this frame on vectors a and b, due to an insufficient rank of the sub-matrix.

IV. Evaluation

In this section we evaluate the decoding complexity and performance of the proposed decoding method.

A. Decoding complexity

The decoding complexity of modified earliest decoding is of order $\mathcal{O}(k^2 \cdot \log k)$, which an improvement on that of Gaussian elimination methods.

The complexity can easily be calculated by considering all the iterations of the decoding algorithm as different levels of a recursion tree. Each level corresponds to the complexity of a single iteration, with the number of levels equal to the number of iterations required for successful decoding. In our case, each iteration requires $\mathcal{O}(k^2)$ calculations. After a single iteration a source packet is obtained, thus after $\mathcal{O}(\log k)$ steps, all source packets will be decoded. Combining these observations, our final result is of order $\mathcal{O}(k^2 \cdot \log k)$.

B. Decoding performance

In contrast to that of avalanche decoding, we do not require the Hamming weight of a received packet's coding

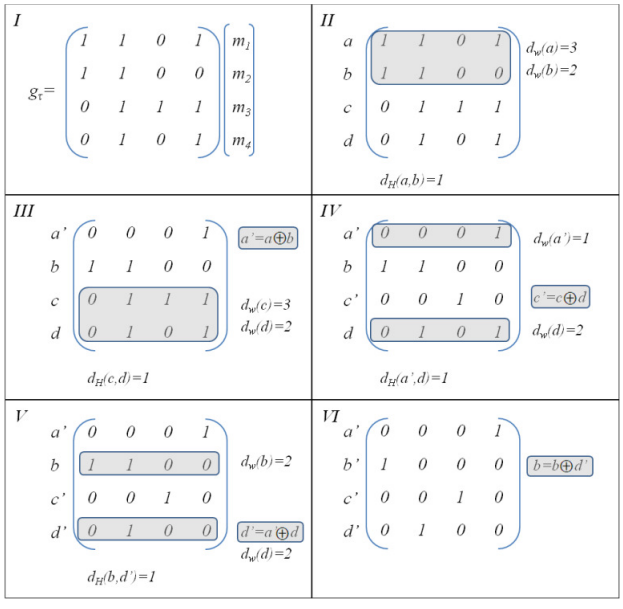


Figure 2. Modified earliest decoding example

vector to be $d_w(x) = 1$ to start decoding. We only require the Hamming distance of two vectors to be $d_H(x, y) = 1$. Thus a linear combination of two such coding vectors results in a coding vector of $d_w = 1$. The same linear combination on the received data packets will result in a data packet that is not linearly combined with any other source packet.

Consider a random linear network coding network where source packets are encoded randomly, independently and are non-zero. Due to the degree distribution degeneration property of random linear network coding networks discussed in [14], there is no benefit in encoding the k independent source vectors $\{X_i\}_{i=1}^{k}$ from $\mathbb{F}_q$ according to a specific distribution. Thus the Gaussian distribution is maintained at the receiver after transmission into the network.

The vectors encoded in the network are then random linear combinations of k possible variables in $\mathbb{F}_q$. We assume that the all-zero vector is present by default in all the decoding buffers of the receiver nodes, because the intermediate nodes do not transmit the all-zero vector. This means that a total of $(2^k - 1)$ vectors may be received.

We evaluate the global encoding vector, g_{t_d}, of a receiver node $t_d \in T$. The receiver obtains a new encoded packet at every transmission instance λ. The decoding of the transmitted message can commence as soon as two packets with global encoding vectors $x, y \in g_{t_d}$ are obtained with $d_H(x, y) = 1$.

Due to the presence of the zero-vector in the decoding buffer, a $d_H(x, y) = 1$ combination can be obtained when any vector with $d_w = 1$ is received. Therefore, the probability of obtaining a $d_H(x, y) = 1$ combination at $\lambda = 1$ is:

$$\rho(\lambda = 1) = \frac{k}{2^k - 1} \tag{4}$$

In order to calculate the probability of decoding to start at instance $\lambda = 2$, we consider the two conditions when it is possible to obtain a combination of two vectors $\{x, y\}$ where $d_H(x, y) = 1$. Firstly, decoding is possible when at least one received vector has $d_w = 1$ with probability:

$$\rho_{d_w=1} = \frac{1}{C}\sum_{j=1}^{2}\binom{2^k - 1 - k}{2 - j}\binom{k}{j}, \tag{5}$$

where $C = \binom{2^k - 1}{2}$.

Secondly decoding can commence when any two received vectors $\{x, y\}$ with $d_w(x) = d_w(y) + 1$ has

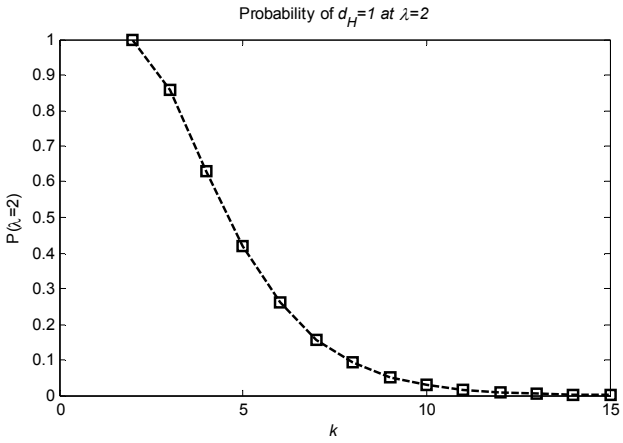


Figure 3. Probability of $d_H = 1$ at $\lambda = 2$.

$d_H(x, y) = 1$ with probability:

$$\rho_{d_H=1} = \frac{1}{C}\sum_{j=2}^{k-1}(k - j)\binom{k}{j}, \tag{6}$$

where $C = \binom{2^k - 1}{2}$.

The overall probability of obtaining a $d_H(x, y) = 1$ combination at $\lambda = 2$ is:

$$\rho(\lambda = 2) = \rho_{d_w=1} + \rho_{d_H=1}. \tag{7}$$

In Fig. 3 we show the probability that a network can commence decoding at $\lambda = 2$ according to (7), for $k = 2, 3, \ldots, 15$. We can see that as k tends to infinity, the probability of $\rho(\lambda = 2)$ decreases.

Next we calculate the probability for obtaining a $d_H(x, y) = 1$ combination at $\lambda > 2$. The probability of obtaining at least two vectors in a selection from λ randomly encoded vectors can be approximated by:

$$\rho_\lambda \cong 1 - (1 - \rho_{\lambda=2})^D, \tag{8}$$

where $D = \frac{\lambda(\lambda-1)}{\lambda/2+1} - \frac{1}{2}$. In Fig. 4 we plotted the probability of decoding to commence at λ for $k = 2, 3, \ldots, 7$.

From Fig. 4 it can be seen that the probability of receiving a combination of vectors with a Hamming distance of 1 in less than $\lambda < k$ transmissions is fairly high. This means that in a practical network, one is likely to be able to start decoding before k packets have been received.

We have not yet been able to explicitly prove that once $k + \delta$ packets have been received, modified earliest

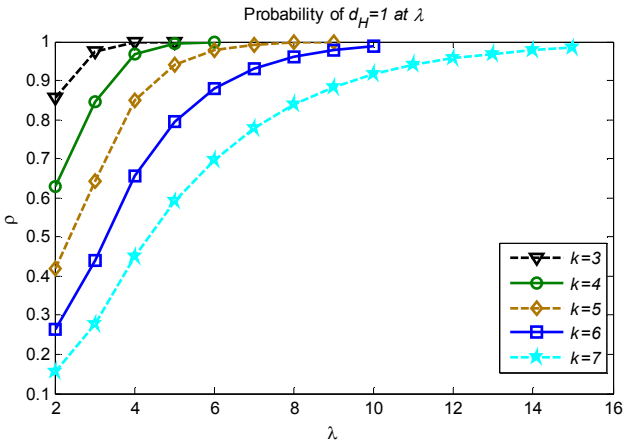


Figure 4. Probability of $d_H = 1$ at λ

decoding will successfully decode all received source packets, for $\delta > 0$. However, in [15] it is shown that k linearly independent vectors can be expected after $\sum_{j=1}^{k} \frac{1}{1-2^{j-1-k}}$ coding vectors have been received. This sum converges to 1.6 equations for $k \gtrsim 10$. Thus, no more than 2 additional packets have to be received for k linearly independent vectors to be obtained from the random linear network coding network [15], providing an upper limit for large k on the number of received vectors required for successful decoding.

It is clear from the above analysis that by using the modified earliest decoding method, decoding can commence at $\lambda < k$ with high probability. If, however, it is unable to decode all received source packets at $\lambda = k + 2$, Gaussian elimination can be performed on the remaining coded packets, thus guaranteeing successful decoding.

V. CONCLUSION

We presented the modified earliest decoding algorithm based on the low complexity concept of avalanche decoding and low decoding delay concept of earliest decoding. The modified earliest decoding algorithm can be applied to systems where a predefined degree distribution is enforced, but is also able to provide decoding performance enhancements when a Gaussian degree distribution is received in a random linear network coding environment.

The modified earliest decoding method is less resource intensive than traditional Gaussian elimination due to a lower decoding complexity of $\mathcal{O}(k^2 \cdot \log k)$. The lower decoding complexity also reduces the decoding time of the algorithm.

From a decoding delay perspective, the modified earliest decoder can start the decoding process as soon as two packets with a Hamming distance $d_H(x, y) = 1$ are obtained. This leads to an improvement of the decoding delay of the network.

Open problems include the calculation of $k + \delta$ with $\delta > 0$ where the start of the decoding algorithm will be guaranteed. Also the determining of the conditions for the method to successfully decode all the received source packets remains an open problem.

REFERENCES

[1] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," *IEEE Int. Symp. Information Theory*, Yokohama, July 2003, p. 442.

[2] H. Wang, J. Goseling, J.H. Weber, "Network coded flooding," Master's thesis, Dept of Telecommunic., Delft University of Technology, June 2009.

[3] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," Allerton Conference on Communication, Control and Computing, Monticello, IL, October 2003.

[4] T. Ho, "Networking from a network coding perspective," PhD Thesis, Massachusetts Institute of Technology, Dept of EECS, May 2004.

[5] D. Silva, F. R. Kschischang, and R. Koetter, "A Rank-Metric approach to error control in random network coding," Proc. Information Theory for Wireless Networks, 2007 IEEE Information Theory Workshop, Solstrand, Norway, July 2007, pp. 1-5.

[6] A. H. Dekker and B. D. Colbert, "Network robustness and graph theory," 27th Australasian Computer Science Conference, Conferences in Research and Practice in Information Technology, vol. 26, V. Estivill-Castro, Ed. 2004.

[7] M. Luby, "LT codes", Proc. The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002, pp. 271-280.

[8] D.J.C. MacKay, "Fountain codes", IEEE Proc. Communications, 2005, vol. 152, pp. 1062-1068.

[9] D.J.C. MacKay, "Information Theory, Inference, and Learning Algorithms", Cambridge University Press, 2003.

[10] M. Champel, K. Huguenin, A. Kermarrec, N. Le Scouarnec, "LT network codes", Institut National de Recherch en Informatique et en Automatique (INRIA), November 2009.

[11] P.A. Chou, Y. Wu, "Network coding for the internet and wireless networks," IEEE Signal Processing Magazine, Sept 200, pp. 77-85.

[12] D.Y. Hu, M.Z. Wang, F.C.M. Lau, Q.C. Peng, "On the design of low compexity decoding (LCD) network codes," Wireless Communications, Networking and Information Security (WCNIS), IEEE International Conference, 2010, pp. 269–273.

[13] F. Fu, V.K. Wei, R. W. Yeung, "On the minimum average distance of binary codes: linear programming approach" Journal on Discrete Applied Mathematics, 2001, pp. 263–281.

[14] A. Hessler, T. Kakumaru, H. Perrey, D. Westhoff, "Data obfuscation with network coding," Computer Communications, Nov 2010.

[15] T. Tirronen, J. Virtamo, E. Hyytia, "Optimizing the degree distribution of LT codes," Master's Thesis, Helsinki University of Technology, Dept of Electrical and Communications Engineering, March 2006.

MODIFIED EARLIEST DECODING IN NETWORKS THAT IMPLEMENT RANDOM LINEAR NETWORK CODING

S. von Solms* and A.S.J. Helberg*

** School of Electric, Electronic and Computer Engineering, North-West University, Potchefstroom Campus, Potchefstroom. E-mail: {sune.vonsolms, albert.helberg}@nwu.ac.za*

Abstract: In this paper we present a formalised description of Modified Earliest Decoding. We analyse through simulation the performance of the method in comparison with Earliest Decoding in networks that implement random linear network coding. We show that Modified Earliest Decoding has a smaller decoding complexity than Earliest Decoding and Gaussian Elimination for small numbers of source packets as well as a smaller decoding delay.

Keywords: Earliest Decoding, Gaussian Elimination decoding, Network Coding, Random Linear Network Coding

1. INTRODUCTION

The decentralised approach to network coding namely random linear network coding (RLNC) allows for a more practical approach to network coding [1]. Random linear network coding employs random and independent coding of received packets at intermediate network nodes.

Due to the employment of RLNC in networks, randomly encoded packets are received at the sink nodes. The sink nodes need to employ a decoding method to successfully obtain the source information. There exist several decoding methods in the literature that can be successfully implemented with RLNC, but generally the sink node may not be able to decode the source packets until an entire block of encoded packets are received [2]. This leads to a decoding delay which is not favourable for delay sensitive networks [2, 3]. Decoding delay can be seen as the elapsed time between the reception of a packet at a receiver node and the decoding thereof [2]. The challenge, therefore, is to find a decoding method with a small decoding delay as well as low decoding complexity.

Gaussian Elimination (GE) is a possible decoding method, but is computationally complex due to the use of matrix inversion and it has a decoding delay equal to the length of time needed by the receiver to collect encoded packets of full rank [4, 5].

Earliest Decoding (ED) is a method developed to decrease the decoding delay of GE. This method entails the use of GE on linearly independent packets of sufficient rank as soon as they are collected by a receiver node. The decoding delay of ED is approximately constant and independent of the number of transmitted source packets, but still employs computationally complex matrix inversion [4 - 6].

In this paper we look at an improvement on ED for the implementation in a network that uses RLNC, called Modified Earliest Decoding (MED). This method is based on ED and shows an improvement on the low decoding delay of ED. Modified Earliest Decoding also reduces the decoding complexity by significantly reducing the use of matrix inversion for decoding. The method of MED was proposed in [7], but it was neither formalised nor analysed. We present a formal algorithm of MED and an analysis through simulation of the decoding delay and complexity of MED.

2. RELATED WORK

A typical network environment where RLNC can be implemented successfully is that of wireless ad-hoc and sensor networks [8]. Wireless sensor networks include scenarios where a block of data or file needs to be transmitted from a single source to a receiver where the intermediate nodes do not require the file. A node in a wireless sensor network is connected to another node in the network when one node is in the coverage of the other node's signal.

It is shown that the random geometric graph (RGG) is a realistic model for a wireless sensor network as it considers the communication distances of nodes [9]. We adopt the notation used in [1, 5] and the graph construction of [10, 11].

Consider an acyclic network which implements RLNC as a random geometric graph $\boldsymbol{\mathcal{G}} = (N, l) = (\mathcal{V}, \mathcal{E})$ with $N = |\mathcal{V}|$. The graph is formed by placing N nodes

uniformly at random on a unit square with communication radius of l. An edge $e = (u, v) \in \mathcal{E}$ exists between two nodes $u, v \in \mathcal{V}$ when the Euclidean distance between u and v is $d(u, v) \leq l$, where the value of l corresponds to the broadcast radius of a node in the wireless network. We assume a symmetric case where all the network nodes have equal transmission power and thus an identical connectivity radius l. The probability p that two nodes u, v are connected is bounded by:

$$\frac{1}{4}(\pi l^2) \leq p \leq \pi l^2. \quad (1)$$

The lower bound is due to the fact that a node can be situated in one of the corners in the unit square. The upper bound is the direct consequence of the communication radius of a node [12].

The wireless sensor network consists of a single source node $s \in \mathcal{V}$ and a set of sink nodes $Z = \{z_1, \dots, z_{|z|}\}$, $Z \subset \mathcal{V}$ with min-cut $(s, z) \geq n$. The data present at the source node, s, is divided into n packets and denoted by

$$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n] \quad (2)$$

where $\boldsymbol{x}_i$ represents the ith source packet from a finite field $\mathcal{F}$ of size q. These source packets are multicast sequentially over the edges $e \in \mathcal{E}$ of network $\boldsymbol{\mathcal{G}}$ to synchronised intermediate nodes $v \subset \mathcal{V}$. This means that the source node does not transmit encoded packets but single source packets one by one.

Each intermediate network node randomly and linearly combines the packets received from its incoming edges e' to form a new encoded packet to be transmitted on its outgoing edges e. A coding vector of length n is included in the header of each outgoing packet. It describes the source packets that have been linearly combined in the transmitted packet.

Each receiver node $z \in Z$ collects a set of $N \geq n$ encoded packets from the network, $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_N]$, where the jth encoded packet is of the form

$$\boldsymbol{y}_j = \sum_{i=1}^{n} g_{ij}\boldsymbol{x}_i, j = 1,2, \dots N \quad (3)$$

where the coefficients $\{g_{ij}\}$ are randomly generated from a finite field $\mathcal{F}_2$ and $\boldsymbol{g}_j$ forms the global coding vector of packet $\boldsymbol{y}_j$. These coding vectors can be represented as the column vectors of a $n \times N$ matrix $\boldsymbol{G}$ [4, 13] where

$$\boldsymbol{X} \times \boldsymbol{G} = \boldsymbol{Y}. \quad (4)$$

The solution of the linear system of equations in (4) decodes the source packets X.

3. RELATED WORK

Next we discuss three known decoding methods that can be implemented in a network that implements RLNC. Earliest Decoding and belief propagation (BP) decoding require alterations to the encoding procedure of the network, where Gaussian Elimination can be implemented without any alterations. The choice of decoding method influences the decoding delay as well as the computational resources at the receiver nodes.

3.1 Gaussian Elimination

Gaussian Elimination is an efficient method for solving a system of linear equations as described in (4). Gaussian Elimination can be performed only when $\boldsymbol{G}$ is of full rank n. Thus the decoding delay of GE equals the time the receiver has to wait in order to collect n linearly independent packets which is proportional to the size of n. Thus the decoding delay of GE increases linearly with the increase of source packets. Gaussian Elimination requires $\mathcal{O}(n^3)$ operations for decoding via matrix inversion which is computationally complex [4, 5]. In a situation of a small number of source messages, GE is an efficient decoding method, but decreases in efficiency as n becomes large.

3.2 Earliest Decoding [4]-[6]

Earliest Decoding performs the same decoding steps as GE but does not require $\boldsymbol{G}$ being of full rank n. Earliest Decoding allows a receiver to perform decoding on a subset of source packets as soon as sufficient information is received, even though the decoding matrix is incomplete. This decoding algorithm is run every time an innovative packet is obtained at the receiver. An *innovative packet* is defined as a packet that increases the rank of $\boldsymbol{G}$. This method enables a receiver to decode a subset of source packets, $\boldsymbol{X}' \subseteq \boldsymbol{X}$ when the global coding vectors in matrix $\boldsymbol{G}' \subseteq \boldsymbol{G}$ can be successfully inverted.

The sequential multicasting of source packets over the network results in $\boldsymbol{G}$ likely to be lower triangular. This means that $\boldsymbol{X}' = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_m], m \leq n$ can be decoded with high probability after the collection of m or $m + \delta$ packets, where δ is small in relation to m [5]. This makes ED practical in a RLNC environment.

Earliest Decoding yields a smaller decoding delay than GE as packets can be decoded by the receiver while still obtaining innovative packets. The decoding delay stays

approximately constant and independent of n [4]. The inherent divide and conquer approach also leads to a faster decoding time, but still requires computationally complex matrix inversion.

3.1 Belief propagation decoding

The decoding method employed for Luby Transform (LT) codes [14] is an iterative process where a sink node first have to determine the degree of each a received packet $\boldsymbol{y}_j$.

Definition 1: The *degree* of a packet indicates the number of source packets linearly combined in the packet or can be seen as the number of non-zero entries in the packet's global encoding vector $\boldsymbol{g}_j$.

The BP decoding process can be described by the following steps [14]:

1. Find an encoded packet, $\boldsymbol{y}_j, 1 \leq j \leq N$, which only contains a single source packet, $\boldsymbol{x}_i, 1 \leq i \leq n$ (i.e. native packet or packet of degree one).
2. Set source packet $\boldsymbol{x}_i = \boldsymbol{y}_j$ and delete $\boldsymbol{y}_j$.
3. Subtract the value of $\boldsymbol{x}_i$ from all the other encoded packets $\boldsymbol{Y} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_{j-1}, \boldsymbol{y}_{j+1}, \dots, \boldsymbol{y}_N]$, $N \geq n$ that contains source packet $\boldsymbol{x}_i$, reducing their degrees.

The reduction of the packet's degrees produces a new native packet with high probability. Repeat process from (1) until all source packets $x_i, 1 \leq i \leq n$ are determined.

To ensure the presence of a native packet each time the process iterates, all packets are encoded according to the Robust Soliton (RS) degree distribution [14].

4. MODIFIED EARLIEST DECODING

Earliest Decoding is successful when m linearly independent packets are present to decode all m source packets. Modified Earliest Decoding [7] allows for an iterative approach to the decoding of source packets.

Modified Earliest Decoding applies the low decoding delay concept of ED but reduces the decoding complexity by significantly reducing the use of matrix inversion. As with ED, MED runs the decoding algorithm every time a new innovative packet is obtained at a receiver node.

The sequential transmission of source packets leads to the scenario where the mth received packet $\boldsymbol{y}_m$ tends to be a linear combination of the first m transmitted source packets $\boldsymbol{X'} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_m], m \leq n$. Due to this lower triangular structure of $\boldsymbol{G}$ it is possible to decode the source packets through a method adopted from the low complexity belief propagation (BP) decoding algorithm of LT codes.

4.1 Decoding in a RLNC environment

In a RLNC network scenario packets are encoded randomly and thus employing a low complexity decoding method, like BP, which requires packets to be from the RS distribution, can be complicated. The lower triangular structure of $\boldsymbol{G}$, however, largely consists of encoded packets where the packets following each other only contains a single additional source packet. A source packet can be decoded when receiving two packets with coding vectors have a Hamming distance of $d_H(\boldsymbol{g}_p, \boldsymbol{g}_q) = 1$.

Definition 2 [15]: The *Hamming distance* between two vectors $\boldsymbol{g}_p, \boldsymbol{g}_q$ is defined as the number of coordinates that they differ and is denoted by

$$d_H(\boldsymbol{g}_p, \boldsymbol{g}_q). \tag{5}$$

Definition 3: The *Hamming weight* of a vector $\boldsymbol{g}_p$ is defined as the number of non-zero coordinates and is denoted by

$$w_H(\boldsymbol{g}_p). \tag{6}$$

In the context of this paper, the Hamming weight of a coding vector is equivalent to the degree of the packet.

The linear combination of two packets with $d_H = 1$ produces a native packet with a coding vector $\boldsymbol{g}_{\boldsymbol{x_i}} = \boldsymbol{g}_p \oplus \boldsymbol{g}_q$ of degree one. This packet contains information regarding a single source packet $\boldsymbol{x}_i$ which is equivalent to a decoded packet. This native packet can now be used for decoding in a similar way as with BP decoding. By linearly combining $\boldsymbol{x}_i$ with other received packets containing $\boldsymbol{x}_i$, the degrees of these packets are reduced. This process is iterated until all the other source packets $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \dots, \boldsymbol{x}_n]$ are decoded.

This MED decoding algorithm is summarised in Algorithm 1.

Algorithm 1: Modified Earliest Decoding

Initialise $\boldsymbol{g_0} = \boldsymbol{0}, \boldsymbol{g_0} \in \boldsymbol{G}$, and $\boldsymbol{y_0} = \boldsymbol{0}, \boldsymbol{y_0} \in \boldsymbol{Y}$
while not all $\{\boldsymbol{x_i}\}_{i=1}^{n}$ are decoded **do**
  Collect $\{\boldsymbol{g_m}\}_{m>0} \in \boldsymbol{G}$ of received packet $\{\boldsymbol{y_m}\}_{m>0} \in \boldsymbol{Y}$
  while $\boldsymbol{g_m}$ contains a native (unit) coding vector $\boldsymbol{g_{x_i}}$ **do**
    $\boldsymbol{g_m} \leftarrow \boldsymbol{g_m} \oplus \boldsymbol{g_{x_i}}$
    update $\boldsymbol{G}$, $\boldsymbol{Y}$
  end while
  determine $\boldsymbol{d_H(g_p, g_q)}, \{\boldsymbol{g_p}, \boldsymbol{g_q}\} \in \boldsymbol{G}, \boldsymbol{q} > p$
  if $\boldsymbol{d_H(g_p, g_q) = 0}$ **do**
    remove $\boldsymbol{g_q}$ from $\boldsymbol{G}$
    update $\boldsymbol{Y}$
  end if

13: **if** $d_H(g_p, g_q) = 1$ **do**
14: $g_{x_i} \leftarrow g_p \oplus g_q$
15: determine $w_H(g_p)$, $w_H(g_q)$
16: remove $g_q, w_H(g_p) < w_H(g_q)$
17: **else** remove $g_p, w_H(g_q) < w_H(g_p)$
18: set $x_i \in X$ equal to $y' \leftarrow y_p \oplus y_q$
19: mark g_{x_i} as a native (unit) coding vector
20: update Y
21: **if** any $\{g_m\}_{m \in N}$ in G contains native g_{x_i} **do**
22: $g_m \leftarrow g_m \oplus g_{x_i}$
23: update G, Y
24: **end if**
25: **end if**
26: **if** no MED is possible, perform ED if possible
27: **end while**

Although the structure of G tends to be lower triangular allowing MED to function successfully, the random encoding of packets do not always guarantee a Hamming distance of 1 between packets. In this instance ED is performed on $G' \subseteq G$ to decode source packets allowing decoding to continue.

4.2 Example

The MED process is illustrated in a small example in Fig. 1. Assume that $n = 4$ and the receiver has obtained 4 encoded packets $Y = [y_1, y_2, y_3, y_4]$ from the network that implements RLNC.

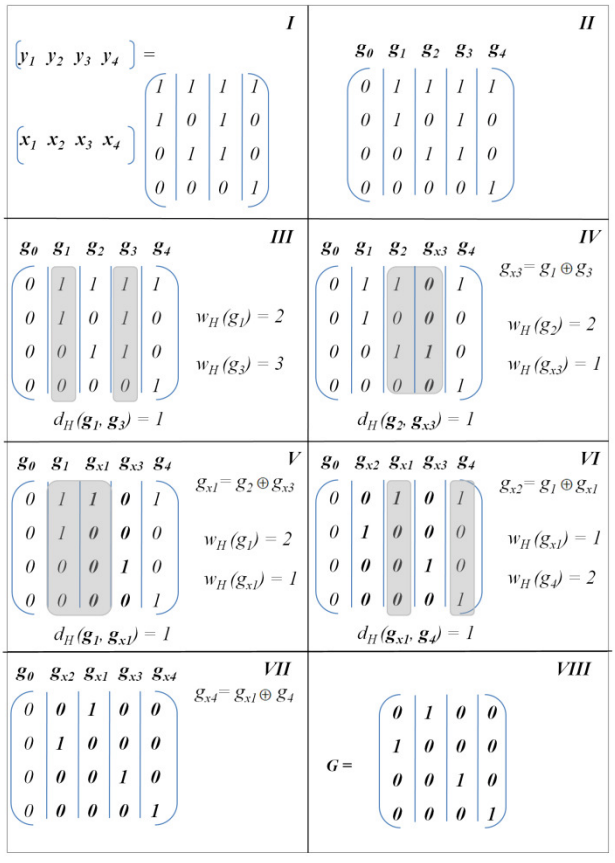


Figure 1. Modified Earliest Decoding example

Frame (I): The global encoding vectors of the received packets are shown. In this example sufficient encoded packets are received before the decoding process starts. This is for clarity purposes for the example. Practically decoding can commence as soon as two packets with a Hamming distance of one are received.

Frame (II): By default a receiver adds the zero-vector as coding vector g_0 to matrix G and then adds the received coding vectors $\{g_i\}_{i=1}^{4}$ to G.

Frame (III): Through evaluation it can be seen that $d_H(g_1, g_3) = 1$. The degrees of the packets are evaluated where $w_H(g_1) = 2$ and $w_H(g_3) = 3$.

Frame (IV): Because $w_H(g_1) < w_H(g_3)$, coding vector g_3 is removed and replaced by native vector $g_{x_3} = g_1 \oplus g_3$. The process starts again where $d_H(g_2, g_{x_3}) = 1$ and $w_H(g_{x_3}) < w_H(g_2)$.

Frame (V): Coding vector g_2 is replaced by native vector $g_{x_1} = g_2 \oplus g_{x_3}$. The next iteration shows that $d_H(g_1, g_{x_1}) = 1$ where $w_H(g_{x_1}) < w_H(g_1)$.

Frame (VI): Coding vector g_1 is removed and replaced by native vector g_{x_2}.

Frame (VI)-(VII): The last undecoded vector is g_4 that can be replaced by native vector g_{x_4} because $d_H(g_{x_1}, g_4) = 1$ and $w_H(g_{x_1}) < w_H(g_4)$.

Frame (VIII): A permuted identity matrix can be seen which shows that all the source packets have been determined and the transmitted data successfully decoded.

Note: In this example only the linear operations performed on the coding vectors are shown. The same operations are performed on the corresponding data packets Y.

From the example it can be seen that MED is a modified version of ED, where MED also forms a sub-matrix from the global encoding matrix to decode. Modified Earliest Decoding, however, eliminates the presence of source packets in the encoded packets that are mutual. The above example cannot be decoded via ED as no sub matrix of G exist that can be inverted. Gaussian Elimination would have been performed after the reception of all the packets.

5. SIMULATION AND RESULTS

In this section we evaluate the decoding performance in the RLNC network environment, as described in Section II, when different decoding methods are implemented at the receiver nodes. We evaluate the decoding delay and decoding complexity for MED in comparison to ED.

5.1 Simulation setup

The network topology is based on that of [9] where the network $G = (N, l) = (\mathcal{V}, \mathcal{E})$ with a single source s and

single receiver z for simplicity. The wireless sensor network is modelled by a random geometric graph (RGG) formed by placing the nodes uniformly at random on a unit square with communication radius of l as described in Section 2. Let $r(s,z)$ be the achievable rate at which s can multicast the source packets reliably to the receiver z.

From the min-cut max-flow theorem, the value of min-cut(s,z) is the upper bound on $r(s,z)$ for z [16]. For a single source single receiver network, the expected value of min-cut(s,z) can be calculated by

$$(N-1)p - \sqrt{(N-1)p(1-p)/\pi} \qquad (7)$$

where the value of p is shown in (1) [10]. In order to ensure the successful transmission of n source packets from s to the receiver node z, the min-cut of the network must be equal to min-cut $(s,z) \geq n$. From (1) and (7) the connectivity radius l and the number of network nodes N are chosen to accommodate the required min-cut value min-cut$(s,z) \geq n$, for varying values of n.

The communication radius l is chosen specifically in each simulation set to ensure a minimum cut between source and receiver node of min-cut $(s,z) \geq n$. If a constructed RGG has a min-cut smaller than the required n, the graph is discarded and a new RGG is generated.

We followed the method of *independent replications* from [17] in order to obtain results which are not affected by different network scenarios. For the simulations we generated 40 random geometric graphs with different seeds. From each random graph we run 5 instances with a different sources and receivers which are randomly chosen. Finally, for each of the sub-instances we ran the simulation 5 times with different seeds. This equates to 1000 Monte-Carlo simulations for each value of n.

The data transmitted by s to the receiver consists of approximately 10000 packets in the finite field $\mathcal{F}_{2^8}$. These packets are divided into n transmission packets $\{x_i\}_{i=1}^{n}$ of size m. A coding vector of length n from finite field $\mathcal{F}_2$ is included in the header of each packet, describing included source packets.

5.2 Decoding delay

As described in Section I, the decoding delay at a receiver is defined as the elapsed time between the reception of an encoded packet and the decoding thereof.

We denote t as the timestep of the simulation when z obtains a new packet from the network. We denote the global rank of the network as R_n, which is equal to the number of source packets n. The rank present at receiver node z at time t is defined as $R_z(t)$. The source packets decodable by node z are defined as *effective packets* and the total number of effective packets at z up to time t is denoted as $E_z(t)$ [18]. The number of effective packets decodable by node z, $E_z(t)$, is upper bounded by the rank present at node z, $R_z(t)$. The value of $R_z(t)$ is in turn upper bounded by the number of packets received by z up until time t.

We ran the MED and ED algorithms at the receiver node for each simulation instance in order to get a fair comparison in decoding delay. Fig. 2 shows the normalised $E_z(t)/R_n$ decoding curves for the MED and ED for $n = 10$, where $N = 250$ and $l = 0.18$. The case for a small n is chosen for Fig. 2 as it clearly illustrates the difference in decoding delay. The curve $R_z(t)/R_n$ shows the normalised value of the rank available at z, which expresses the total number of source packets possibly decodable at time t. This curve gives the upper limit of decoding for any system at time t.

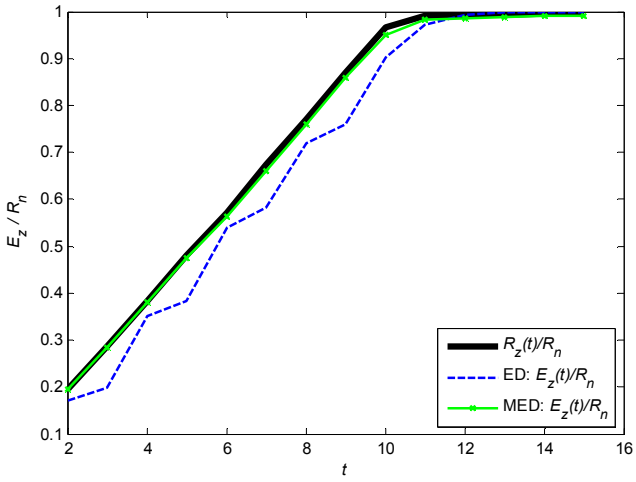


Figure 2: Normalised decoding delay of MED and ED for $n = 10$

It can be seen in Fig. 2 that MED produces a larger number of effective packets at z when innovative packets are still being received than compared to ED. This means that the MED method is able to decode more source packets at time t than the ED method, resulting in a smaller decoding delay. The graph further shows variable decoding delay for ED and constant decoding delay MED which may be of advantage for certain applications.

The decoding delay of ED is independent of n and remains approximately constant [4, 5]. From simulation in [4] it was shown that the algorithmic decoding delay of ED is often only in the order of a few source packets, much smaller than n. Therefore, the decoding delay observed in Fig. 2 would continue to be in the order of a few source packets even for larger values of n.

The same observation can be made for MED as this method also decodes subsets of a few source packets

much smaller than n. The decoding method of MED for larger values of n continues to decode small subsets of source packets, independent of n, therefore producing a decoding delay in the order of a few source packets.

Thus for large values of n, the decoding delay of MED remains approximately constant and an improvement on that of ED, as can be seen in Fig. 2.
The decoding delay of MED is also upper bounded by that of ED, because when no packets with $d_H = 1$ are available the MED algorithm reverts to ED.

5.3 Decoding complexity

In addition to decoding delay another important characteristic of a decoding method is its decoding complexity. We determined the decoding complexity of ED and MED for $10 \leq n$ by calculating the number of arithmetic operations for both decoding methods.

Earliest Decoding consist of two steps namely *forward elimination* and *backward substitution*. The number of arithmetic operations for the forward elimination step is approximately

$$\sum_{i=1}^{u-1}(u-i) + \sum_{i=1}^{u-1}(u-i)^2 \qquad (8)$$

divisions and multiplications/additions respectively, where $u < 1 \leq n$ is the size of the subset of source packets decoded in each step. The backward substitution step requires

$$\sum_{i=1}^{u-1}(u-i) \qquad (9)$$

multiplications/additions.

For the MED algorithm the zero vector is added to $\boldsymbol{G}$, which leads to $u+1$ packets per subset. The $u+1$ packets are compared to each other which can require a maximum of

$$u + (u-1) + \cdots + 1 = \frac{u(u+1)}{2} \qquad (10)$$

arithmetic operations, where $u \leq 1 \leq n$ is the size of the subset of source packets decoded in each step. After these comparisons a single source packet is decoded and eliminated from the other packets in the block, which requires a maximum of

$$u - 1 \qquad (11)$$

operations per decoded packet.

We use the abovementioned formulae to determine the decoding complexities of MED in comparison to ED through simulation. We define the normalised decoding complexity as the ratio of the number of operations for successful MED to the number of operations for successful ED [19].

Fig. 3 shows the decoding complexity advantage of MED over ED for varying values of n. It can be seen that the decoding complexity of MED is lower than that of ED. It is clear from the analysis of arithmetic operations in (8) – (11) that both methods are of the same order of complexity. There, however, is an improvement of approximately 30% on the number of arithmetic operations required for successful decoding for MED over ED, as shown in Fig. 3.

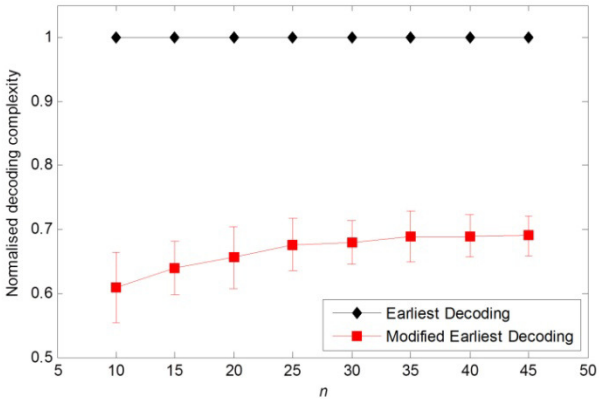


Figure 3: Normalised decoding complexity of MED over ED

From the results obtained it can be seen that MED renders favourable results of an improved decoding complexity. MED also renders a smaller decoding delay when compared to ED.

6. CONCLUSION

The Modified Earliest Decoding algorithm is based on the low complexity belief propagation decoding concept of LT codes and low decoding delay concept of Earliest Decoding.

In this paper we formalised the MED process in Algorithm 1. This algorithm shows how the Hamming distances between coding vectors can be used to obtain native packets for successful decoding.

The performed simulations show a lower decoding delay as well as a lower decoding complexity of MED over ED. When no coding vectors of $d_H = 1$ are present in $\boldsymbol{G}$ and MED cannot be performed, ED is used and therefore the decoding delay of MED is upper bounded

by that of ED for large n.

Earliest Decoding showed a significant improvement on decoding delay and complexity in comparison to Gaussian Elimination [4, 5] when implemented in a RLNC network environment. Our improvement over ED further improves the performance over GE which is the predominant decoding method in RLNC network environment.

7. 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 Proc. IEEE Int. Symp. on Inform.Theory (ISIT), Yokohama, pp. 442, Jul. 2003.

[2] J. K. Sundararajan, D. Shah and M. Medard, "Online network coding for optimal throughut and delay – the three receiver case," in Proc. Int. Symp. on Inform. Theory and its Applic., Aukland, New Zealand, 2008.

[3] R. A. Costa, D. Munaretto, J. Widmer and J. Barros, "Informed network coding for minimum decoding delay," in Proc. IEEE Int. Conf. on Mobile Ad-hoc and Sensor Systems, Atlanta, Georgia, Sept 2008.

[4] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in Proc. Allerton Conference on Communication, Control and Computing, Monticello, IL, October 2003.

[5] P. A. Chou, Y. Wu, "Network coding for the internet and wireless networks," IEEE Signal Processing Magazine, pp. 77-85, Sept 2003.

[6] H. Wang, J. Goseling, J. H. Weber, "Network coded flooding," M.S. thesis, Dept. of Telecommunications, Delft University of Technology, Delft, June 2009.

[7] S. von Solms and A. S. J. Helberg, "Modified Earliest Decoding for Random Network Codes," in Proc. The 2011 Int Symp. on Network Coding (NetCod), Beijing, China, 2011.

[8] D. Petrovic, K. Ramchandran and J. M. Rabaey, "Overcoming untuned radios in wireless networks with network coding," IEEE Trans. on Inf. Theory, vol.52, no. 6, pp. 2649-2657, 2006.

[9] A. Ramamoorthy, J. Shi, and R. Wesel, "On the capacity of network coding for random networks," IEEE Trans. on Inf. Theory, vol. 51, no. 8, pp. 2878–2885, 2005.

[10] H. Wang, P. Fan and K. B. Letaief, "Maximum Flow and Network Capacity of Network Coding for Ad-hoc Networks," IEEE Trans. on Wireless Commun. vol. 6, no. 12, pp. 4193-4198, 2007

[11] C. Avin, "Random Geometric Graphs: An Algorithmic Perspective," Ph.D dissertation, University of California , Los Angeles , 2006

[12] S. A. Aly, V. Kapoor, J. Meng, and A. Klappenecker, "Bounds on the network coding capacity for wireless random networks," in *Proc.*3rd Workshop on Network Coding, Theory, and Applications, San Diego, CA, U.S.A., 2007.

[13] T. Ho, "Networking from a network coding perspective," PhD Thesis, Massachusetts Institute of Technology, Dept. of EECS, May 2004.

[14] M. Luby, "LT codes", in Proc. The 43rd Annual IEEE Symp. on Foundations of Computer Science, pp. 271-280, 2002.

[15] F. Fu, V. K. Wei, R. W. Yeung, "On the minimum average distance of binary codes: linear programming approach" Journal on Discrete Applied Mathematics, pp. 263–281, 2001.

[16] R. Koetter and M. Medard, "Beyond routing: an algebraic approach to network coding," in Proc. Twenty-First Annual Joint Conf. of the IEEE Computer and Communications Societies, vol. 1, pp. 122-130, 2002.

[17] D. Goldsman and G. Tokol, "Output analysis: output analysis procedures for computer simulations," in Proc. The 32nd Conf. on Winter simulation (WSC), pp. 39-45, 2000

[18] L. Lima, D. Ferreira, and J. Barros, "Topology matters in network coding," Springer Telecommunications systems, pp. 1-11, 2011.

[19] S. Kim, K. Ko and S. Chung, "Incremental Gaussian Elimination Decoding of Raptor Codes over BEC," IEEE Commun. Letters, vol. 14, no. 4, April 2008.

Evaluation of Decoding Methods for Random Linear Network codes

Suné von Solms, *Student Member, IEEE,* Albert S. J. Helberg, *Member, IEEE*

***Abstract*— In this paper we present a network configuration for Random Linear Network Coding networks to obtain non-strict lower triangular generator matrices at the receiver nodes. Further we present and utilise a mathematical model to evaluate the decoding performances of Earliest Decoding and Modified Earliest Decoding. We show that Modified Earliest Decoding provides a noticeable advantage in the decoding delay of the transmitted data above Earliest Decoding as Modified Earliest Decoding is more resilient to packet erasures.**

***Index Terms*— Earliest Decoding; Modified Earliest Decoding; Network Coding; Random Linear Network Coding.**

I. Introduction

The use of Gaussian Elimination (GE) for decoding in Random Linear Network Coding (RLNC) networks has the disadvantage of high computational complexity and delay, which can outweigh the practical advantages that RLNC offers, especially for large blocks of information [1], [2], [3]. Thus the effectiveness of RLNC in a practical setting is dependent on the complexity and decoding delay of the implemented decoding algorithm [4].

Earliest Decoding (ED) was introduced in [1] as a decoding method to improve the decoding delay in a RLNC network. ED is a more cost effective decoding method as it decodes the source data in smaller subsections, thus offering practical decoding delays as it enables receiver nodes to decode source symbols while innovative encoded packets are still being received from the RLNC network. ED entails the use of GE on linearly independent packets of sufficient rank as soon as the packets are collected by a receiver node.

Modified Earliest Decoding (MED) was developed in [5], [6] as an improvement to ED for implementation in the same RLNC environment. MED is adapted to implement the iterative decoding principle of belief propagation used in fountain codes in order to eliminate the use of computationally complex matrix inversion.

The decoding efficiency of ED and MED is dependent on a specific lower triangular structure of the generator matrix obtained by a receiver node. It is stated in [1], [7] that the causality of computations in a RLNC network can lead to received packets having a lower triangular structure. This means that the ith received packet is a linear combination of the first i source symbols with high probability. Thus, for each generation to be successfully decoded with ED and MED, not only must enough linearly dependent packets be received, but preferably in lower triangular form to reduce decoding delay.

In the literature where ED was presented and implemented [1], [7] it was only stated that the lower-triangular form of the received packets are due to network causality. To the best of our knowledge, no coding method or network configuration was presented on how this lower triangular structure was obtained.

In this paper we present a practical network configuration for RLNC networks to obtain the lower triangular structure of encoded packets, so that ED and MED can be successfully implemented. We show how to obtain the lower triangular structure of the generator matrix without altering the encoding algorithm at intermediate network nodes. We illustrate how the causality of the random network can render packets with a lower triangular structure if the source transmits packets in a defined manner.

A mathematical model of the lower triangular generator matrix structure is developed to calculate the decoding performances of ED and MED in a RLNC network scenario where the received packets forms a non-strict lower triangular generator matrix. Through the use of the mathematical model we also determine the influence of an erased packet on the decoding efficiency of ED and MED as a packet erasure can compromise the lower triangular structure of the generator matrices at the receiver nodes.

II. Related Work

A. Earliest Decoding

Earliest Decoding consists of the same decoding steps as GE, but does not require the generator matrix, $\boldsymbol{G}$, at the receiver node to be of full rank n, where n is the size of the generation. Earliest Decoding allows a receiver to perform decoding on a subset of source packets as soon as sufficient information is received, even though the decoding matrix is incomplete. The decoding algorithm is run every time an innovative packet is obtained at the receiver.

The decoder determines the number of undecoded source symbols at the node (number of unknowns) as well as the

Manuscript received November 20, 2012. This work was completed with funding from the Telkom Centre of Excellence at the North-West University, Potchefstroom Campus.

S. von Solms and A. S. J. Helberg are with the School of Electric, Electronic and Computer Engineering, North-West University, Potchefstroom Campus, Potchefstroom, South Africa. (E-mail: sune.vonsolms@nwu.ac.za, albert.helberg@nwu.ac.za)

number of linearly independent coding vectors (number of linear equations). When a receiver node has collected $1 \leq i \leq n$ linearly independent packets containing information regarding i undecoded source symbols, the decoding of the i source symbols is possible, through matrix inversion. When a source symbol is decoded, it is linearly combined with all the new incoming packets in order to eliminate the decoded symbol from the new packets received from the network. This means that decoding of a subset of source symbols, $\boldsymbol{X_{sub}} \subseteq \boldsymbol{X}$, can take place through the inversion of a sub matrix $\boldsymbol{G_{sub}} \subseteq \boldsymbol{G}$ while innovative packets are still being received and the decoding matrix $\boldsymbol{G}$ is incomplete [1], [7], [8].

ED is developed for the practical network scenario where the received packets have a lower triangular structure [1]. The decoding delay and complexity of ED is approximately constant and independent of the number of transmitted source symbols, [1], [7], [8].

B. Modified Earliest Decoding

MED applies the low decoding delay concept of ED but reduces the decoding complexity by eliminating the use of matrix inversion. As with ED, MED also runs the decoding algorithm every time a new innovative packet is obtained at a receiver node. In a network environment where the global encoding vectors of the received packets form a generator matrix which is roughly lower triangular, MED can improve the decoding delay compared to ED.

Definition 1 [9]: We define the Hamming distance, denoted by $d_H\left(\boldsymbol{g}(e_i), \boldsymbol{g}(e_j)\right)$, between two coding vectors $\boldsymbol{g}(e_i), \boldsymbol{g}(e_j)$ as the number of coefficients $[g_1(e), g_2(e), \dots, g_n(e)]$ in which they differ.

Definition 2: We define the Hamming weight of the coding vector $\boldsymbol{g}(e_i)$ as the number of non-zero coefficients $[g_1(e), g_2(e), \dots, g_n(e)]$ and is denoted by $w_H(\boldsymbol{g}(e_i))$. A packet with a coding vector with $w_H(\boldsymbol{g}(e_i)) = 1$ is called a native packet as it only contains a single source symbol.

The lower triangular structure of $\boldsymbol{G}$ enables a receiver node to obtain native packets when receiving two packets with global encoding vectors with a Hamming distance of $d_H\left(\boldsymbol{g}(e_i), \boldsymbol{g}(e_j)\right) = 1$. With the linear combination of two such packets, a packet is obtained which contains only a single source symbol, called a native packet. This native packet can now be used for decoding other encoded packets. By linearly combining it with other encoded packets containing the decoded source symbol, the Hamming weights of the encoded packets are reduced. This process is iterated to decode the other source packets.

III. Practical Network Configuration

The lower- triangular form of the generator matrix received at the receiver nodes is what enables the receiver nodes of the network to perform ED. To the best of our knowledge the manner in which to obtain a lower triangular generator matrix at the receiver nodes is not discussed in the literature. It was only stated that the lower triangular form of the packets are due to the causality of the network.

We present a network configuration in order to obtain generator matrices at sink nodes that resemble a lower triangle in order to successfully implement ED as well as MED. All the intermediate network nodes implement RLNC and the structure of the generator matrix is obtained through specified encoding at the source node of the network.

A. Network causality

A causal network can be described as a directed acyclic graph where the data present at each non-source node is a function of the data available at its parent nodes [10]. We assume that all edges of the non-cyclic network $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$ have unit capacity but may have multiple edges between nodes in order to model different edge capacities and that information flows over the edges in zero time. Due to causality of such a network, it can be assumed with high probability that the source symbols contained in the first encoded packets transmitted over the network would be the first to be received at the sink nodes.

From this we can deduce that a lower triangular structure of the generator matrix, $\boldsymbol{G}$, can be obtained at the receiver nodes if the n source symbols $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n] \in \mathbb{F}_q$ are transmitted sequentially over the network. This means that at transmission opportunity $t = 1$, the source would transmit an encoded packet containing symbol $\boldsymbol{x}_1$, and at transmission opportunity $t = 2$ an encoded packet containing symbol $\boldsymbol{x}_2$, and so on. This would lead to a sink node sequentially receiving innovative packets possibly containing only a single new source symbol.

B. Network configuration

We consider a network that can be represented by a directed acyclic graph $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$, where the source node $s \in \mathcal{V}$ contains n source symbols $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n] \in \mathbb{F}_q$. In this instance only a single generation of size n is considered, but can be easily extended to multiple generations. Every time the source is presented with a transmission opportunity, an encoded packet $y(e)$ would be constructed from a linear combination of the source symbols where

$$y(e) = \beta_1(e)\boldsymbol{x}_1, \beta_2(e)\boldsymbol{x}_2, \dots, \beta_n(e)\boldsymbol{x}_n$$
$$y(e) = \sum_{i=1}^{n} \beta_i(e)\boldsymbol{x}_i. \quad (1)$$

where $\boldsymbol{\beta}(e) = [\beta_1(e), \beta_2(e), \dots \beta_i(e)]$ are generated from finite field $\mathbb{F}_2$ and forms the local encoding vector of packet $y(e)$ at the source node. These coefficients of $\boldsymbol{\beta}(e)$, however, are not randomly chosen, but selected based on the

transmission time step t. When $t = i$, the local encoding coefficients of the ith encoded packet $y(e)$ would be constructed as follows:

$$\beta_j(e) = \begin{cases} \{1,0\} & \text{if } j < i \\ 1 & \text{if } j = i \\ 0 & \text{if } j > i \end{cases} \quad (2)$$

The random selection of local encoding coefficients $\beta_j(e), j < i$ is to ensure that source symbols are randomly included in other encoded packets so that a source symbol would not be lost due to a packet erasure. The inclusion of the source symbol when $j = i$ is to ensure that each encoded packet contains only one new source symbol, as no symbols are further included when $j > i$. Thus the source effectively transmits encoded packets that resemble a strict lower-triangular matrix, where the ith transmitted packet contains a linear combination of the first i source symbols.

When the source node is presented with a transmission opportunity, these encoded packets are multicast over the outgoing edges of the source node and RLNC is performed on the packets received at the intermediate nodes. As the first packets transmitted over the network contained the first source symbols, the causality of the network allows that the source symbols that entered the network first have a high probability of arriving first at the receiver nodes.

The encoded packets $y(e)$ collected by the receiver nodes $Z = \{z_1, \dots, z_{|z|}\}, Z \subset \mathcal{V}$ can be written as linear combination of the source symbols

$$y(e) = \sum_{i=1}^{n} g_i(e)\, \boldsymbol{x}_i \quad (3)$$

where $\boldsymbol{g}(e) = [g_1(e), g_2(e), \dots, g_n(e)]$ is the global encoding vector over $\mathbb{F}_2$ of each packet $y(e)$.

Due to the causality of the network, we can assume that the receiver nodes receive encoded packets in a sequential manner where the packets first transmitted by the source node over the network are the first packets to be included in the encoded packets $\boldsymbol{Y}$ collected by the receiver nodes. Thus the global encoding vectors of the collected packets $\boldsymbol{Y}$ at each receiver node form with high probability the coding vectors of a non-strict lower triangular coding matrix $\boldsymbol{G}$, where

$$\boldsymbol{G} \times \boldsymbol{X}^T = \boldsymbol{Y}^T. \quad (4)$$

As the finite field over which coding is performed is $\mathbb{F}_2$, $\boldsymbol{G}$ can be considered as a binary matrix.

This network configuration enables the receiver nodes of the network to obtain with high probability packets whose global coding vectors resemble a lower- triangular matrix. This means that in a lossless scenario the ith received packet $y(e_i)$ received by a receiver node tends to be a linear combination of the first i transmitted source packets $\{\boldsymbol{x}_p\}_{p=1}^{i}$.

The lower triangular structure of $\boldsymbol{G}$, however, is not guaranteed, as the structure is dependent on the connectivity and structure of the network. All intermediate nodes encode packets randomly. In a decentralised network the length of max-flow paths may vary. These factors may influence the sequential structure of the packets, resulting in source symbols transmitted in early packets arriving later than symbols transmitted after them. Therefore it is possible for the sink to collect an encoded packet $y(e_i)$ containing new source symbols $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ before collecting a packet containing $\boldsymbol{x}_i$. This means that the matrix may contain non-zero elements in the second, third or higher, diagonals.

An example of such a non-strict lower triangular $\boldsymbol{G}$ matrix obtained form a RLNC network where the source symbols are sequentially encoded and transmitted as discussed in this section, is depicted in Fig. 1 for $n = 100$. The first 11 received packets are enlarged and shown in Fig. 2 (b) to illustrate the occurrence of packets that are received out of order due to network factors like varying lengths of max-flow paths.

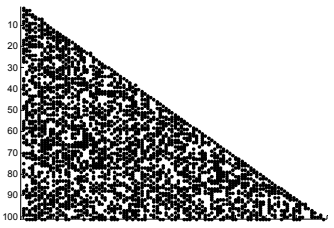
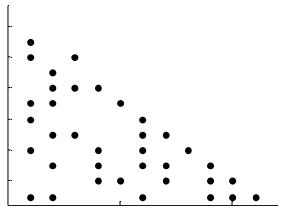


Figure 1: (a) Random non-strict lower triangular generator matrix



(b) First 11 collected packets of generator matrix

It can be seen that this network setup allows us to obtain a $\boldsymbol{G}$ matrix that resembles the form of the $\boldsymbol{G}$ matrix described in [*1*] so that ED and MED can be successfully implemented.

IV. Model of Generator Matrices

In this section we evaluate the structure of the generator matrices obtained through the use of the above network configuration. We model the generator matrices in terms of non-zero entries in the diagonals and through Monte Carlo simulations determine the practical probabilities for the non-zero entries.

A. Generator matrix model

In the network configuration described in Section III, the finite field over which coding is performed is $\mathbb{F}_2$ which allows all the elements of matrix $\boldsymbol{G}$ to be either equal to 1 or 0. $\boldsymbol{G}$ can be considered as a binary matrix where the entries $\{g_{ij}\}$ of $\boldsymbol{G}$ are distributed according to a Bernoulli distribution where

$$\{g_{ij}\} = \begin{cases} 1 & \text{with probability } P_{ij} \\ 0 & \text{with probability } (1 - P_{ij}) \end{cases}. \quad (5)$$

Due to the fact that the generator matrix $\boldsymbol{G}$ has a lower triangular structure, we can assume that the probabilities P_j of the occurrence a non-zero element in a specific diagonal are equal to each other, as shown in Fig. 2.

Therefore not all the entries have unique distributions, but entries in the same diagonal can form a collection with equal probabilities for the occurrence of a non-zero element. We represent P_1 as the probability of receiving a 1 on the main diagonal of the matrix. This translates to the ith received packet containing source symbol $\boldsymbol{x}_i$. P_2 is the probability of receiving a 1 on the second diagonal of the matrix, thus

receiving source symbol $\boldsymbol{x}_{i+1}$ in the ith packet. Due to the fact that the source node randomly includes source symbols $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{i-1}$ for the ith packet, we consider the probability for the occurrence a non-zero element below the diagonal as equal, with value P_x.

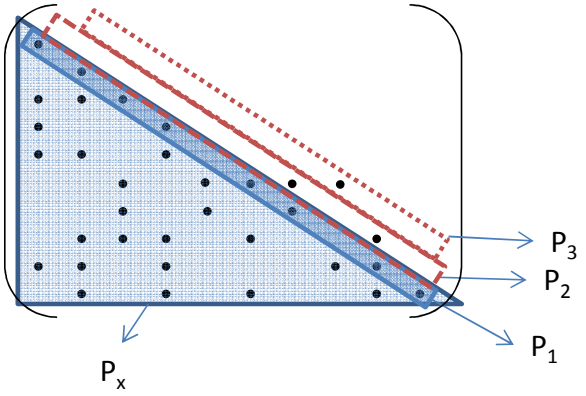


Figure 2: Lower Triangular matrix with diagonal probabilities

Therefore we can simplify (7) and state probabilities for all diagonals where

$$\{g_{ij}\} = \begin{cases} 1 \text{ with probability } P_m \\ 0 \text{ with probability } (1 - P_m) \end{cases}, \forall\, i = j - m + 1 \quad (6)$$

is the probability for non-zero elements in the mth diagonal.

B. *Establishing probabilities of non-zero matrix elements*

The network configuration developed in Section III can enable us to calculate the probabilities of obtaining non-strict lower triangular generator matrices at receiver nodes of a RLNC network. In order to do so, the probabilities of obtaining non-zero elements in the diagonals of the generator matrices constructed at the receiver nodes must be determined.

To obtain probabilities that would accurately represent a practical RLNC network, the network configuration described in Section III is simulated in a RLNC network environment. Through Monte-Carlo analysis we determine values of P_x, P_1, P_2 etc. to model the behaviour of a practical RLNC network.

1) *Simulation Setup*

The network topology used for these simulations can be described by a network $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$ with a single source s, multiple receivers $Z = \{z_1, \dots, z_{|z|}\}, Z \subset \mathcal{V}$ and intermediate nodes which implement RLNC. The number of nodes in the network is $R = |\mathcal{V}|$. We base our experimental setup on that of [11] where random geometric graphs (RGG) are used to model a network implementing RLNC. We selected the network parameters to approximate the practical network considered in [1].

The network is modelled by a random geometric graph formed by placing the R nodes uniformly at random on a unit square, each node with communication radius of l. An edge $e = (v, w) \in \mathcal{E}$ exists between two nodes $v, w \in \mathcal{V}$ when the Euclidean distance between v and w is $d(v, w) \leq l$. We assume a symmetric case where all the network nodes have equal transmission power and thus an identical connectivity radius l.

Let $r(s, Z)$ be the achievable rate at which s can multicast the source packets reliably to the receivers. From the min-cut max-flow theorem, the value of min-cut(s, Z) is the upper bound on $r(s, Z)$ [12]. In order to ensure the successful transmission of n source packets from s to Z, the connectivity radius l and the number of network nodes R are chosen to accommodate the required min-cut value min-cut$(s, Z) \geq n$, for varying values of n. If a constructed RGG has a min-cut smaller than required, the graph is discarded and a new RGG is generated.

The data transmitted by the source node s to Z consists of approximately $hn = 10000$ source symbols in the finite field $\mathbb{F}_{2^8}$. The source symbols are divided into h generations of sizes varying from $n = 35$ to $n = 100$, where the ith generation contains packets $\{x_{(i-1)n+j}\}_{j=1}^{n}$ where RLNC at intermediate nodes is performed over $\mathbb{F}_2$. The min-cut (s, Z) from source to receiver nodes must support the transmission of generations of sizes varying from $n = \{35{,}100\}$.

We followed the method of independent replications [13] in order to obtain results which are not affected by different network scenarios. For the simulations we generated 40 random geometric graphs with different seeds. For each random graph we run 5 instances with different sources and receivers which are randomly chosen. Finally, for each of the sub-instances we ran the simulation 5 times with different seeds. This equates to 1000 Monte-Carlo simulations.

2) *Simulation Results*

For each replication, the coding vectors of all the innovative packets received at each sink node were placed in a matrix $\boldsymbol{G}$ and evaluated. As we use diagonals to characterise the generator matrices, we determine the probability of non-zero elements for each diagonal in $\boldsymbol{G}$.

It was found that even when the value of n is changed, the diagonal probabilities remained approximately the same. This shows that the selection of l and N to sustain the required min-cut value for varying values of n is done correctly. The obtained probabilities for the various diagonals of matrix $\boldsymbol{G}$ are summarised in Table 1.

Table 1: Probabilities of non-zero elements in $\boldsymbol{G}$

	Diagonal	Probability
$\boldsymbol{P_1}$	1 (main)	0.94
$\boldsymbol{P_2}$	2	0.05
$\boldsymbol{P_3}$	3	0.016
$\boldsymbol{P_i, i \geq 4}$:	≥ 4	< 0.011
$\boldsymbol{P_x}$	below main	0.5

The probabilities shown in Table 1 show a relationship with the encoding of the local encoding coefficients for each packet at the source node shown in (2). The probability of obtaining non-zero elements in the main diagonal, P_1, is not 1 and the possibility of obtaining non-zero entries above the main diagonal is not 0. In decentralised networks, the length of

max-flow paths may vary which can influence the sequential flow of packets through the network resulting in source symbols transmitted in early packets arriving later than symbols transmitted after them. Also, as coding is performed over $\mathbb{F}_2$, a source symbol in a packet can be eliminated when linearly combined with another packet that contains the same symbol. These factors can cause a packet to have an entry in the second diagonal and possibly a zero entry in the main diagonal.

3) Discussion

The mathematical model describing the generator matrices at the receiver nodes correlates with the practical RLNC network scenario when we use the probabilities obtained from the practical RLNC network are used in the model. This mathematical model enables the evaluation of the decoding efficiency of ED and MED in the developed network configuration.

In the evaluation of decoding methods in the RLNC network configuration developed in Section III, we only consider the probabilities of P_x, P_1 and P_2, thus assuming $P_i = 0$, where $, i \geq 3$. It can be seen that the probability of obtaining non-zero entries in the third diagonal is $P_3 = 0.016$, and above the 3rd diagonal less than 1%.

The exclusion of P_3would cause the mathematical model to differ from the practical network in the scenarios where a non-zero entry is obtained in the 3rd diagonal. In the practical network evaluated, a non-zero entry is found in the 3rd diagonal approximately 1.6% of the time. Thus the exclusion of P_3 in our mathematical model is an inaccurate representation of the practical network 1.6% of the time. The inclusion of P_3, however, would greatly complicate the mathematical model. We selected to build a model which is uncomplicated with the knowledge that it would only accurately represent a practical network 98.4% of the time. The practical networks represented form a specific category of networks, thus a change in a network parameter like topology would have a larger influence on the mathematical model than the exclusion of the probabilities above the 2nd diagonal.

V. Decoding Performance in Lossless Network Scenario

In this section we evaluate the decoding performance of ED and MED in a RLNC network with the network configuration presented in Section III. The mathematical model presented in Section IV is used to randomly and independently generate encoded packets with the probabilities of non-zero entries as shown in Table 1. One thousand generations of various sizes of n were generated with different seeds and were decoded through the use of ED and MED.

We assume a lossless environment to determine the decoding efficiency of ED as well as MED in the RLNC network where the network configuration enables the receiver nodes of the network to obtain packets whose global coding vectors resemble a non-strict lower triangular matrix as shown in Fig. 1. As the encoded packets are generated, they are decoded through the use of ED and MED. In this network no innovative packet is lost, thus the ith received packet $y(e_i)$ received by a receiver node tends to be a linear combination of the first i transmitted source packets $\{\boldsymbol{x}_p\}_{p=1}^{i}$ with high probability. The median of the decoding delay of ED and MED for $n = 10$ are shown in Fig. 3 where t denotes the timesteps when a new packet is collected for decoding.

Definition 3: Decoding delay is defined as the elapsed timeslots between the reception of a packet at a receiver node and the decoding thereof [14]. In a time slot, receiver nodes experience a decoding delay of 1 when an encoded packet is successfully received, but not decoded [15].

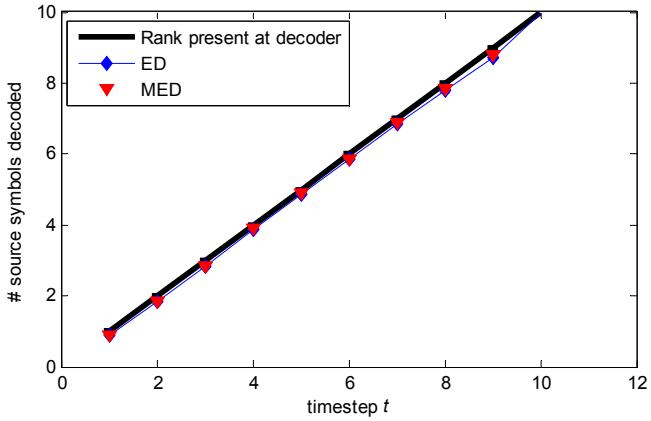


Figure 3: Decoding delay for ED and MED in lossless environment

From Fig. 3, it can be seen from the '*Rank present at the decoder*' curve that the receiver node collects an innovate packet at each timestep. It can be seen that both ED and MED are able to decode with high probability a single source symbol with the collection of each now encoded packet. Thus the advantage of the lower triangular structure of $\boldsymbol{G}$ can be seen as the decoding delay of both decoding methods is approximately zero.

VI. Decoding Performance in Practical Network Scenario

In a practical RLNC network, packets can be lost with probability ρ_e due to channel interference from external devices, changes in topology, link failures or other detrimental network conditions. RLNC networks generate redundant encoded packets at the intermediate nodes which can be used for erasure control against packet loss [16], [3]. When a packet is lost within a generation, the receiver node can still obtain sufficient linearly independent packets for decoding if it waits a certain amount of time to collect enough packets. This characteristic makes RLNC robust to erasures as successful decoding at receiver nodes do not depend on receiving packets that contain specific information, but on receiving enough linearly independent packets [17].

In the network environment presented in Section III, however, the efficiency of decoding is improved by obtaining packets that contain new information in a structured manner.

These packets are collected by the receiver nodes in a systematic order which enables the construction of generator matrices of roughly lower triangular structures. This structure enables efficient decoding but can be compromised by the erasure of a packet. Although decoding would still be possible at the end of the generation when sufficient linearly independent packets are collected, the advantage of improved decoding delay via ED and MED can be lost.

A. *Influence of an erasure on lower triangular* $\boldsymbol{G}$ *matrix*

In the RLNC network environment where received packets form a non-strict lower triangular matrix of size n, we label the packets received before the erasure with indices $1, \dots, k$. Thus the lost packet would be labeled $(k+1)$ and the packets received afterward labeled $(k+2), \dots, n$. Assume that ρ_e is the probability of the erasure of an innovative packet occurring in the transmission channel of a specific sink node. The number of successful innovative packet receptions that can be expected until the first erasure is:

$$E = \frac{1-\rho_e}{\rho_e} \tag{7}$$

As the first k packets obtained by the receiver nodes include a single undecoded source symbol with high probability, packet $y(e_{k+1})$, when not received due to an erasure, possibly carried a new undecoded source packet which is not included in the generator matrix of the receiver $\boldsymbol{G}$. Thus the following packet, $y(e_{k+2})$, can potentially introduce two new undecoded source packets to a sink node. If packets subsequent to $y(e_{k+1})$ continue to introduce a single undecoded source symbol each, $\boldsymbol{G}$ would consist of n source symbols and only $(n-1)$ linearly independent packets. Due to the redundancy introduced in the RLNC network, innovative packet $y(e_{n+1})$ can be obtained which would render $\boldsymbol{G}$ to be of full rank. The loss of a packet can be seen as the main diagonal of the $\boldsymbol{G}$ matrix shifting to the right, as illustrated in Fig. 4.

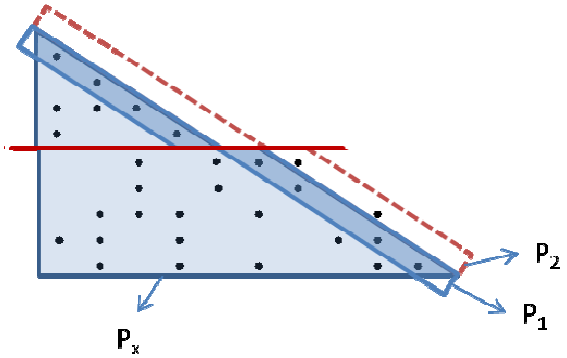


Fig. 4: Non-strict lower triangular matrix $\boldsymbol{G}$ with a single packet erasure

It can be seen from Fig. 4 that the first k received packets are still packets with probabilities of non-zero entries in the main and second diagonals as shown in Table 1. After a packet is erased, it can be seen that the probabilities of the non-zero entries in the different diagonals have shifted to the right. These probabilities are shown in Table 2.

Table 2: Probabilities of non-zero elements in $\boldsymbol{G}$ after erased packet

	Diagonal	Probability
$\boldsymbol{P_1}$	2	0.94
$\boldsymbol{P_2}$	3	0.05
$\boldsymbol{P_i, i \geq 3}$:	≥ 3	0
$\boldsymbol{P_x}$	main and below	0.5

From Fig. 4 and Table 2 it can be seen that the probability of packet $y(e_{k+2})$ introducing at least two undecoded source symbols to the receiver is high. The probability of having a non-zero entry on the main and second diagonal has changed to $P_x = 0.5$ and $P_1 = 0.94$ respectively. Packet $y(e_{k+3})$ can introduce a new source symbol with probability $P_1 = 0.94$ as this now is the probability of a non-zero entry in the second diagonal. In addition, packet $y(e_{k+2})$ can now introduce three undecoded source symbols to the receiver as it may contain a non-zero entry in the third diagonal with probability $P_2 = 0.05$.

B. *Decoding performance in practical network*

In this section we consider the scenario where the sink node does not obtain all the systematically encoded packets. The mathematical model presented in Section IV is used to randomly and independently generate encoded packets with the probabilities of non-zero entries shown in Tables 1 and 2. One thousand generations of various sizes of n were generated with different seeds, where single packet erasure is guaranteed in each generation. As encoded packets are generated, they are presented to a MED and ED decoder. The median of the decoding delay of ED and MED for $n = 100$ are shown in Fig. 5 where t denotes the timesteps when a new packet is collected for decoding.

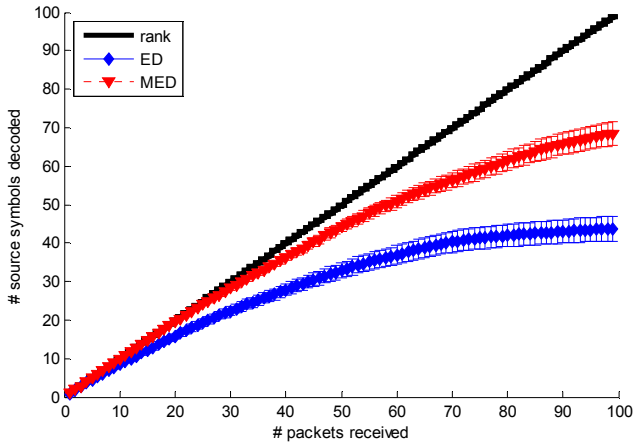


Figure 5: Decoding delay for ED and MED in the presence of packet loss

A RLNC network generates redundant packets which would enable the decoding of all the source symbols are possible at the end of the generation. We are, however, interested in the decoding performance when packets of the lower triangular

structure are still received. Thus the number of decoded packets are only calculated until timestep $t = 99$.

It can be seen from the above figure that MED is more robust to packet loss than ED. Clearly, when a packet is erased, ED cannot continue to decode packets with low decoding delay. The MED algorithm can decode a large portion of source symbols in the presence of packet erasure without erasure protection. In Fig. 4 it can be seen that the number of source symbols decoded by MED is higher than the number of source symbols decoded by ED for the same packets received. MED decoded approximately 70% of the transmitted packets in the presence of an erased packet, where ED only decoded approximately 40%.

VII. Conclusion

In this paper we presented a network configuration of which the causality of the practical RLNC network would result in receivers obtaining packets of a non-strict lower triangular structure. This generator matrix structure enables the implementation of Earliest Decoding as well as Modified Earliest Decoding. We analysed the structure of the lower triangular generator matrices and presented a mathematical model to model the probability of obtaining a generator matrix of such a structure. This model was used to calculate the decoding delay of ED and MED.

In lossless networks, the practical performance of ED and MED are approximately equal. In the case of a single packet loss, the lower triangular structure of global encoding vectors of the received packets are compromised which affects the decoding performance of both methods. The decoding probabilities for both methods were determined and it was shown that MED has more resilience to packet loss in this network scenario and can decode packets as they are collected, producing a lower decoding delay than ED.

Network coding was developed specifically for multicast networks. Multicast networks are widely employed for content streaming applications on the internet and private networks. These applications are often delay sensitive. From the evaluation of MED it can be seen that MED can provide a noticeable advantage in the decoding delay of the data above ED since it is more resilient to packet erasures.

References

[1] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, vol. 41, pp. 40-49, 2003.

[2] H. Shojania and B. Li, "Parallelized Progressive Network Coding With Hardware Acceleration," *Proceedings of the 2007 IEEE International Workshop on Quality of Service*, pp. 47-55, 2007.

[3] S. Yang and R. W. Yeung, "Large file transmission in network-coded networks with packet loss: a performance perspective," *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*, pp. 117:1--117:5, 2011.

[4] D. Y. Hu, M. Z. Wang, F. C. M. Lau, and Q. C. Peng, "On the design of low compexity decoding (LCD) network codes," *Proceedings of the IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, p. 269–273, 2010.

[5] S. von Solms and A. S. J. Helberg, "Modified Earliest Decoding for Random Network Codes," *Proceedings of the 2011 International Symposium on Network Coding (NetCod)*, 2011.

[6] S. von Solms and A. S. J. H. S.J., "Error correction for resource limited random network coding networks," *Proceedings: 2011 IEEE Africon, Livingstone, Zambia*, 2011.

[7] P. A. Chou and Y. Wu, "Network Coding for the Internet and Wireless Networks," *Signal Processing Magazine, IEEE*, vol. 24, pp. 77-85, 2007.

[8] J. H. Weber, "Network coded flooding," Master's Thesis, Dept. of Telecommunications, Delft University of Technology, 2009.

[9] R. W. Yeung, "On the minimum average distance of binary codes: linear programming approach," *Journal on Discrete Applied Mathematics*, p. 263–281, 2001.

[10] D. Heckerman, "A Bayesian Approach to Learning Causal Networks," *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pp. 285-295, 1995.

[11] T. Ho, "Networking from a network coding perspective," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[12] R. Koetter and M. Medard, "Beyond routing: an algebraic approach to network coding," *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2002*, vol. 1, pp. 122-130, 2002.

[13] D. Goldsman and G. Tokol, "Output analysis: output analysis procedures for computer simulations," *Proceedings of the 32nd conference on Winter simulation*, pp. 39-45, 2000.

[14] P. Sadeghi, R. Shams, and D. Traskov, "An Optimal Adaptive Network Coding Scheme for Minimizing Decoding Delay in Broadcast Erasure Channels," *EURASIP Journal of Wireless Communications and Networking*, vol. 2010, 2010.

[15] P. S. D. and R. Koetter, "Adaptive network coding for broadcast channels," *Proceedings of the Workshop on Network Coding, Theory, and Applications (NetCod '09)*, pp. 80-85, 2009.

[16] H. Wang, J. Liang, and J. Kuo, "Overview of robust video streaming with network coding," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, pp. 36-50, 2010.

[17] C. Fragouli, J.-Y. L. Boudec, and J. Widmer, "Network coding: an instant primer," *Computer Communication Review*, vol. 36, pp. 63-68, 2006.