Chapter 4: Implicit Error Detection

Contents

In this chapter we investigate an implicit error detection method. This method collects the packets implicitly formed by the RLNC process and constructs a generator matrix that can be used for error detection. Through analysis we determine the error detection capability of the code for different network topologies. This error detection method does not cost the network any additional resources when implemented.

4.1 Introduction

In Chapter 3 we discussed the need for error and erasure correction in RLNC networks to ensure network reliability. Accordingly, it was stated that the RLNC network acts as an erasure protection code because it forms redundant packets in the network that can be used for decoding when packets are lost in the network [**8**], [**10**], [**24**], [**33**]. Non-innovative packets collected by the receiver nodes are discarded as they convey no new information to the receiver nodes [**5**], [**19**].

In an RLNC environment, FEC codes can be implemented as an outer code to enable network error correction. This outer code adds additional parity symbols at the source which are functions of the original source symbols, as discussed in Chapter 3.

Thus, in an RLNC network where FEC is implemented, additional parity symbols are generated by the source node which are functions of the original source symbols; and the random encoding/recoding of packets in an RLNC network produces further non-innovative packets which are discarded by the receiver nodes when not required.

In [**50**] a technique was presented where possible errors can be detected without the addition of an outer code at the source node of the network $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$. In this case, the source only transmits the k original source symbols over the network with a *min-cut*$(s, Z) \geq n > k$. In Chapter 3 we discussed the fact that intermediate nodes generate redundant encoded packets when the number of edges out of the intermediate node is larger than that of the network min-cut [**31**], [**32**]. Thus, the intermediate nodes may generate redundancy so that the receivers can obtain k innovative packets and $(n-k)$ additional packets. Although only k packets are required for the decoding of the k source packets, the additional $(n-k)$ packets are not discarded, but used for error detection. These additional packets, implicitly formed in the RLNC network, provide the receiver with parity information that can be used for the detection of erroneous packets. However, the method in [**50**] was only presented and simulated for limited network scenarios and was not investigated further.

Our main contribution in this chapter is the presentation of an improvement to the method developed in [**50**]. We evaluate this method by deriving an analytical expression to describe the reception of packets in an RLNC network. This expression is used to calculate the number of additional packets required to guarantee single error detection. To this end, we conduct simulations to evaluate the influence of network topology on the scheme to find the correlation between the developed mathematical model and the network environment. Lastly, we show that the implementation of implicit error detection can offer several advantages in resource limited networks.

4.2 Network error correction

In Chapter 3 we stated that the error correction capability of a code $\mathcal{C}$ is determined by the minimum Hamming distance of the code words generated by the generator matrix $\boldsymbol{G_{FEC}}$. A

$(n, k, 2t + 1)$ code is t-error correcting or $2t$ error detecting when the minimum distance of the code words generated by $\boldsymbol{G}$ is $d_{min} = 2t + 1$, where $2t \leq (n - k)$ [**28**].

It is possible for a linear code $\mathcal{C}$ to have several distinct generator matrices $\boldsymbol{G_{FEC}}$ for the mapping of the k source packets to n coded packets. A $k \times n$ matrix is a valid generator matrix $\boldsymbol{G_{FEC}}$ when

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

Knowledge of the structure of a valid generator matrix for an (n, k) FEC code is necessary in order to construct and use the implicit error detection method successfully. This is subsequently presented.

4.3 Implicit error detection

In a network $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$ with min-cut $(s, Z) \geq n$ where RLNC is implemented, as presented in Chapter 2, error detection is possible without the addition of an outer code at the source node. We introduce an implicit error detection method where the receiver nodes are able to detect errors although no outer code is implemented. This method is based on the method first presented in [**50**].

4.3.1 Encoding

In this method the source node $s \in \mathcal{V}$ also divides the source data into k symbols $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k]$ over finite field $\mathbb{F}_q$ where $k < n$, but the source packets are not encoded with an outer code. Only a single generation is considered in this network framework in order to illustrate the method, but can be easily extended to multiple generations. As described in Chapter 2, each intermediate node $v \in \mathcal{V}$ transmits an encoded symbol $y(e)$ on its outgoing edges, e, which is a linear combination of the symbols received on incoming edges e'. These encoded packets transmitted over the network and received by the receiver nodes can be seen as linear combinations of the original source symbols $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k$,

$$y(e) = \sum_{i=1}^{k} g_i(e)\, \boldsymbol{x}_i \tag{4.1}$$

where $\boldsymbol{g}(e)$ is the global encoding vector over $\mathbb{F}_q$ of packet $y(e)$which describes $y(e)$in terms of the source symbols $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_k$.

In a network with min-cut $(s, Z) \geq n$, the network can support the independent transmission of n independent source symbols, although only k source symbols were transmitted. Thus, the random encoding characteristic of RLNC causes intermediate nodes to generate redundant packets so that each receiver node can obtain with high probability at least n encoded packets where k are innovative and the rest non-innovative.

4.3.2 Matrix construction

Assume that a receiver node collected $N \geq n$ encoded packets from the network. From the N collected packets, receiver nodes evaluate the encoding vectors of the packets and select only n packets. The global encoding vectors of these n packets are used as row vectors to form an $n \times k$ generator matrix $\boldsymbol{G_{IEC}}$

$$\boldsymbol{G_{IEC}} = \begin{pmatrix} g_1(e_1) & g_2(e_1) & \cdots & g_k(e_1) \\ g_1(e_2) & g_2(e_2) & \cdots & g_k(e_2) \\ \vdots & \vdots & \ddots & \vdots \\ g_1(e_n) & g_2(e_n) & \cdots & g_k(e_n) \end{pmatrix}. \tag{4.2}$$

The generation matrix $\boldsymbol{G_{IEC}}$ shows how the message packets $y(e_i)$ corresponding to the selected row vectors in $\boldsymbol{G_{IEC}}$ are linearly encoded. All the encoded packets $\boldsymbol{Y} = [\boldsymbol{y}(e_1), \boldsymbol{y}(e_2), \ldots, \boldsymbol{y}(e_n)]$ can be described in terms of the original source symbols through the use of $\boldsymbol{G_{IEC}}$, where

$$\begin{pmatrix} g_1(e_1) & g_2(e_1) & \cdots & g_k(e_1) \\ g_1(e_2) & g_2(e_2) & \cdots & g_k(e_2) \\ \vdots & \vdots & \ddots & \vdots \\ g_1(e_n) & g_2(e_n) & \cdots & g_k(e_n) \end{pmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_k \end{bmatrix} = \begin{bmatrix} y(e_1) \\ y(e_2) \\ \vdots \\ y(e_n) \end{bmatrix} \tag{4.3}$$

$$\boldsymbol{G_{IEC}} \times \boldsymbol{X^T} = \boldsymbol{Y^T}$$

The process shown in (4.3) can be related to two encoding procedures.

1. *RLNC encoding*. In an RLNC network, each encoded packet $\boldsymbol{y}(e_i)$ received from an incoming edge e_i of a receiver is effectively a linear combination of the original source symbols $\boldsymbol{X}$. The linear combination of the source symbols in the packet $\boldsymbol{y}(e_i)$ is described by its global encoding vector$\boldsymbol{g}(e_i)$. This global encoding vector is a row vector $\boldsymbol{g}(e_i)$ in the generator matrix $\boldsymbol{G_{NC}}$.

2. *FEC encoding*. With the implementation of an FEC code at the source node, each coded symbol $\boldsymbol{u}_i$ generated by the code, as described in Section 3.2.2, is a linear combination of the original source symbols $\boldsymbol{X}$. The linear combination used to obtain each coded symbol $\boldsymbol{u}_i$ is determined by the generation matrix of FEC code $\boldsymbol{G_{FEC}}$. This linear combination is described by the column vector of $\boldsymbol{G_{FEC}}$.

Thus, we can compare $\boldsymbol{G_{IEC}}$ to $\boldsymbol{G_{FEC}}$ in (3.1) and $\boldsymbol{G_{NC}}$ in (3.3) shown in Chapter 3. It can be seen that $\boldsymbol{G_{IEC}}$ has the same dimensions as an FEC generator matrix, like $\boldsymbol{G_{FEC}}$, but is obtained through the RLNC network, like $\boldsymbol{G_{NC}}$. In all the cases, however, the generator matrix describes how the encoded packets or coded symbols are linearly combined from the original source symbols $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k]$.

Hence, it can be seen that we constructed a generator matrix through the random encoding process of an RLNC network, and not a predetermined algorithm. The generator matrix $\boldsymbol{G_{IEC}}$

therefore describes the linear encoding of the received packets. So how does our constructed matrix differ from that of $\boldsymbol{G_{NC}}$ as it is obtained from the coding vectors of randomly encoded packets?

At the start of this section, we stated that the receiver node collected $N \geq n$ encoded packets from the network, but only used n of these packets to construct $\boldsymbol{G_{IEC}}$. These n packets were constructed in such a way that the row vectors of $\boldsymbol{G_{IEC}}$ have a specific minimum distance, which is a characteristic of a generator matrix of an FEC code. Thus, matrix $\boldsymbol{G_{IEC}}$ has the same characteristics of a generator matrix of an FEC code $\mathcal{C}$, $\boldsymbol{G_{FEC}}$, as discussed in Section 4.2. This matrix can be used to detect possible errors in the network, just like the FEC matrix described in Chapter 3.

4.3.3 Error detection

Through the use of traditional linear error detection decoding, as described in Chapter 3, the receiver node can detect possible errors. From $\boldsymbol{G_{IEC}}$ a valid $(n-k) \times n$ parity check matrix $\boldsymbol{H}$ can be constructed, where $\boldsymbol{H} \times \boldsymbol{G_{IEC}} = \boldsymbol{0}$. The receiver has to multiply $\boldsymbol{H}$ and $\boldsymbol{Y}$ to obtain a syndrome vector $\boldsymbol{z}$. When the syndrome $\boldsymbol{z} \neq \boldsymbol{0}$, it indicates that an error has occurred in the network. By contrast, when $\boldsymbol{z} = \boldsymbol{0}$, no error has occurred in the network and the original source symbols can be retrieved through decoding.

4.3.4 Error detection capability

In section 4.3.2 we stated that the receiver nodes select the global encoding vectors of n received packets to form the generator matrix $\boldsymbol{G_{IEC}}$. These global encoding vectors must have certain characteristics in order for the receiver node to use matrix $\boldsymbol{G_{IEC}}$ for error detection.

The error detecting capability of linear code is determined by the structure of $\boldsymbol{G_{IEC}}$. For an error detection and correction code $\mathcal{C}$, there are several distinct $n \times k$ generator matrices. An $n \times k$ matrix is a valid generator matrix $\boldsymbol{G_{IEC}}$ when:

1. It has a rank of k, i.e. it has k linearly independent rows.
2. Its rows vectors are valid code words in code $\mathcal{C}$, meaning that the minimum Hamming distance of the rows corresponds to the error correction capability of the code $\mathcal{C}$.
3. When the minimum Hamming distance $d_{min} \geq 2$, the generator matrix contains two linearly independent sets of encoded packets of size k and $(n-k)$ respectively. This means that the matrix must contain k rows that are linearly independent of each other, as well as another set of $(n-k)$ packets that are linearly independent of each other.
4. All the data symbols must be present 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 $d_{min} \geq 2$.

Thus, when we aim to construct an FEC generator matrix from an RLNC network, the matrix must adhere to the characteristics mentioned above. This means that the global encoding vectors of

n received packets must adhere to these characteristics. If such a matrix can be constructed, we are able to implement error detection without the need for an outer code at the source node.

As already stated, the global encoding vectors of n received packets must adhere to these characteristics. However, the global encoding vectors of the received packets are all encoded randomly through RLNC in a decentralised network environment. Thus, the packets are encoded at random and the structure of the packets obtained by the receivers is random, where no specific structure of global encoding vectors can be guaranteed. In the following section we evaluate the probability of obtaining packets which can be used to implement implicit error detection as well as calculate the number of expected additional packets.

4.4 Mathematical model

Traditionally, it is assumed that the size q of the finite field $\mathbb{F}_q$ over which the coding is performed is large. A large finite field would almost guarantee encoded packets to be linearly independent of one another, as discussed in Chapter 2. This would enable each receiver node to nearly always decode the source data once only k packets have been collected [**5**]. Performing coding over large field sizes, however, leads to an increase in computational complexity, which can be unsuitable in practical network scenarios [**61**], [**62**]. As stated in Chapter 2, research was done on RLNC over small finite field sizes to provide an acceptable probability of linear independence at low computational cost [**6**], [**18**], [**60**], [**61**]. It was shown in [**8**], [**60**] that there is a high probability of receiver nodes obtaining sufficient innovative packets if coding is performed randomly, independently and over a sufficiently large finite field relative to the size of the network. Thus sufficient innovative packets can be obtained for successful decoding when $\mathbb{F}_q$ is small and the network sufficiently large with a small excess of non-innovative packets [**61**], [**63**]. For this study coding would only take place over finite field $\mathbb{F}_q$ where $q = 2$.

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 [**69**] was used to determine the probability of a receiver node obtaining k linearly independent packets within the first $N \geq k$ packets received. The model considered a network where the packets collected by the sink nodes are received uniformly at random and encoded independently. In large enough networks with high connectivity and sufficient minimum cut, the encoding at intermediate nodes and the collection at the receiver nodes can be modelled as such a random selection. Using the same model of random packet reception, we construct a mathematical model (analytical expression) to do the following:

1. Analyse P: the probability of constructing a valid $n \times k$ generator matrix $\boldsymbol{G_{IEC}}$ after $N \geq n \geq k$ received packets.
2. Calculate the number of additional packets required to construct a valid generator matrix $\boldsymbol{G_{IEC}}$.

In order to calculate the probability of successful generator matrix construction, we need to know the key characteristics of a valid generator matrix, where $d_{min} \geq 2$. These characteristics are described in Section 4.3.4.

We consider a network represented by a directed acyclic graph $\boldsymbol{\mathcal{G}} = (\mathcal{V}, \mathcal{E})$ as discussed in Chapter 2, where the source node $s \in \mathcal{V}$ divides the source data into k packets. These source packets are multicast over the edges $e \in \mathcal{E}$ of network $\boldsymbol{\mathcal{G}}$ to a set of receiver nodes $Z = \left\{z_1, \ldots, z_{z_{|z|}}\right\}$, $Z \subset \mathcal{V}$ as discussed in Chapter 2. We assume that the receiver nodes each receive $N \geq n$ randomly encoded packets from the network. In order to derive the exact expression for P, we need to calculate the following probabilities:

1. $P_k(N, k)$: the probability of obtaining k linearly independent packets within the first $N \geq n$ packets collected, which is calculated in Section 4.4.1
2. p_d: the probability of $(n - k)$ remaining packets being linearly independent
3. p_s: the probability of $(n - k)$ remaining packets containing all the source symbols, which are both calculated in 4.4.2.

4.4.1 Probability of obtaining k linearly independent packets within the first $N \geq k$ packets

In [**61**] and [**69**] an analysis was performed to determine the probability of obtaining k linearly independent packets from the first $N \geq k$ packets. In Sections 4.4.1 to 4.4.3 the analysis is extended to incorporate the probability of obtaining additional $(n - k)$ packets in order to successfully construct a valid generator matrix $\boldsymbol{G_{IEC}}$.

Case for $\boldsymbol{N = k}$

First, we calculate the probability of a receiver node obtaining k linearly independent packets from the first k packets obtained, that is, $N = k$. Over the finite field $\mathbb{F}_2$, there are 2^k possible packets that can be constructed with 2^k different coding vectors. The zero vector cannot be selected, because the network only encodes non-zero packets, thus there are only $2^k - 1$ possible selections.

The first collected packet can have any coding vector except for the zero vector. The probability of collecting one of the other $2^k - 1$ possibilities equals

$$\frac{2^k - 1}{2^k - 1} = 1. \tag{4.4}$$

For the second selection to be innovative, the receiver cannot collect the vector selected in the previous round or the zero vector. Thus the probability of collecting another linearly independent vector is:

$$\left(\frac{2^k-2}{2^k-1}\right)=\left(1-\frac{1}{2^k-1}\right). \tag{4.5}$$

For the third collection the third vector has to be different to the previous two vectors, as well as the linear combination of the two selected vectors. This means that there are 2^2 linearly dependent vectors. So at each selection, the receiver cannot collect the vectors previously chosen or any of the linear combinations of them. The number of linearly dependent vectors that is generated from i selected vectors is equal to 2^i. This means that after $i-1$ linearly independent equations, the probability of selecting the ith linearly independent random vector is

$$\left(\frac{2^k-2^{i-1}}{2^k-1}\right)=\left(1-\frac{2^{i-1}-1}{2^k-1}\right) \tag{4.6}$$

We denote ρ_k as the probability of obtaining k linearly independent packets. From the above calculations, we can calculate the probability of selecting k linearly independent vectors in k selections. Thus, the probability of obtaining k linearly independent vectors from $N=k$ selections is:

$$\rho_k=\prod_{i=1}^{k}\left(\frac{2^k-2^{i-1}}{2^k-1}\right)=\prod_{i=1}^{k}\left(1-\frac{2^{i-1}-1}{2^k-1}\right) \tag{4.7}$$

Figure 4.1 displays the above probability ρ_k for variable sizes of $N=k$:

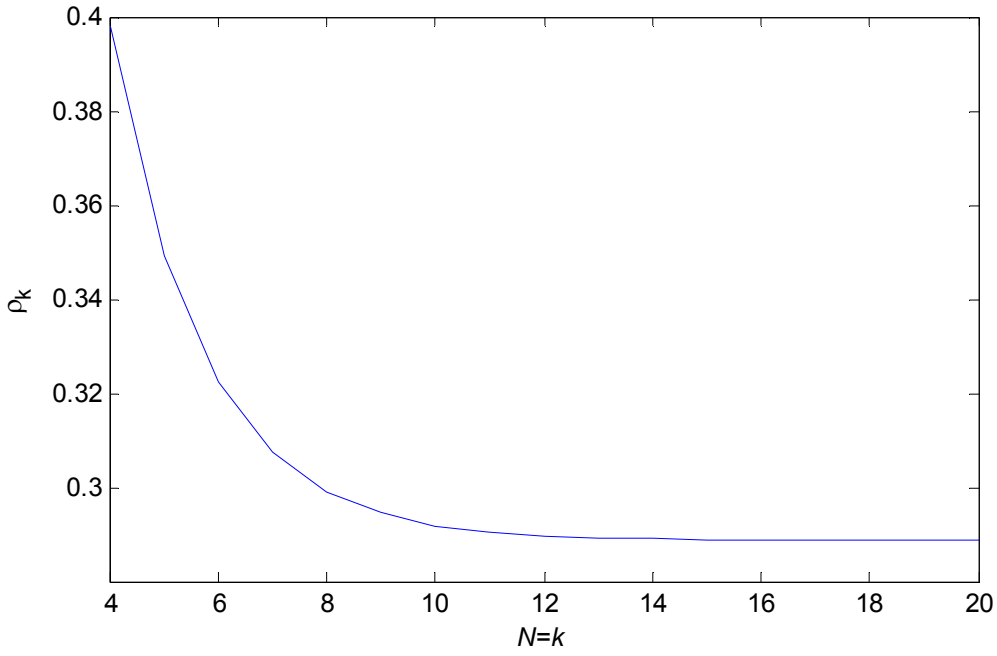

Figure 4.1: Probability of obtaining k innovative packets

Case for $\boldsymbol{N = k + 1}$

From the results obtained above we can calculate the probability of success when $N = k + 1$ packets are collected at a receiver node. As above, the first packet collected can contain any coding vector except the zero vector. The second packet collected allows for the collection of a single non-innovative packet with probability $\left(\frac{2}{2^k-1}\right)$. If a linearly dependent packet is collected, then all the remaining $k - 1$ packets collected must be linearly independent. If the second packet is linearly independent of the first packet with probability $\left(1 - \frac{1}{2^k-1}\right)$, then the receiver must collect $k - 2$ linearly independent packets from the next $k - 1$ collections. The ith packet collected can be linearly dependent with probability $\left(\frac{2^{i-1}-1}{2^k-1}\right)$ or linearly independent with probability $\left(1 - \frac{2^{i-1}-1}{2^k-1}\right)$. Through the iteration of this process the probability, $P_k(N, k)$, of obtaining k linearly independent packets from $N = k + 1$ collections is

$$P_k(N, k) = \rho_k \times \sum_{j=0}^{k} \frac{2^j - 1}{2^k - 1}. \tag{4.8}$$

Case for $N \geq k$

From the structure of (4.8), the probability of obtaining k innovative packets from $N = k + 2$, packets can be calculated as:

$$\begin{aligned} P_k(N, k) &= \rho_k \times \sum_{r1=0}^{k} \frac{2^{r1} - 1}{2^k - 1} \sum_{r2=r1}^{k} \frac{2^{r2} - 1}{2^k - 1} \\ &= \rho_k \times \frac{1}{(2^k - 1)^2} \times \sum_{r1=0}^{k} 2^{r1} - 1 \sum_{r2=r1}^{k} 2^{r2} - 1. \end{aligned} \tag{4.9}$$

From the previous calculations we are able to deduce the probability of success $P_k(N, k)$ for $N \geq k$. The formula can be compared to that of (4.9), where we now have $(N - k)$ summations, where

$$\begin{aligned} P_k(N, k) &= \rho_k \times \sum_{r1=0}^{k} \frac{2^{r1} - 1}{2^k - 1} \sum_{r2=r1}^{k} \frac{2^{r2} - 1}{2^k - 1} \dots \sum_{r(N-k)=r(N-k-1)}^{k} \frac{2^{r(N-k)} - 1}{2^k - 1} \\ &= \rho_k \times \frac{1}{(2^k - 1)^{(N-k)}} \times \sum_{r1=0}^{k} 2^{r1} - 1 \sum_{r2=r1}^{k} 2^{r2} - 1 \dots \sum_{r(N-k)=r(N-k-1)}^{k} 2^{r(N-k)} - 1 \end{aligned} \tag{4.10}$$

In [**61**] it is shown that equations in the form of (4.10) can be reduced through the use of Gauss coefficients to

$$P_k(N,k) = \prod_{i=0}^{k-1}\left(1 - \frac{1}{2^{N-i}}\right) \text{ for } N \geq k. \tag{4.11}$$

Equation (4.11) shows the cumulative distribution function of the probability of receiving k linearly independent packets, given the reception of $N \geq k$ packets under random linear network coding. From the above calculations, we illustrate the probability of obtaining k innovative packets after the reception of k, $k+1$ and $k+2$ packets, respectively, in Figure 4.2.

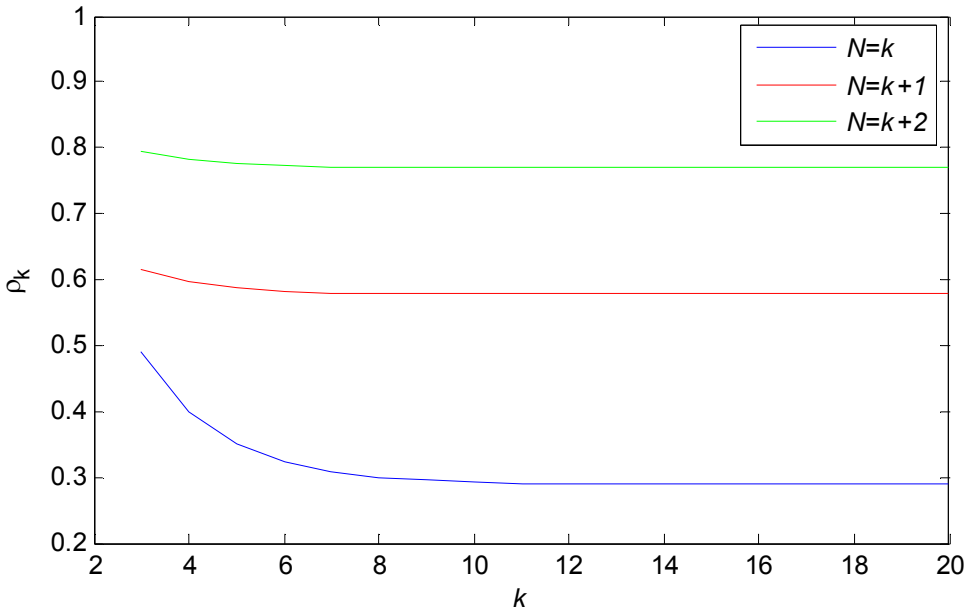

Figure 4.2: Probability of obtaining k innovative packets from N collected packets

4.4.2 Expected number of additional packets required to obtain k linearly independent packets

Next, we calculate the expected number of additional packets required by a receiver node to successfully obtain k linearly independent packets. Note that the probability of requiring i additional packets to collect the next legitimate packet is geometrically distributed is

$$Pr(i,\rho) = \rho \times (1-\rho)^{i-1}, i = 1,2 \ldots \tag{4.12}$$

where ρ is the probability that we succeed in collecting the next legitimate packet. The expectation of (4.12) is equal to

$$\sum_{i=1}^{\infty} i \times Pr(i,\rho) = \sum_{i=1}^{\infty} iP(1-P)^{i-1} = \frac{1}{\rho_k} \tag{4.13}$$

The number of additional packets required to find the mth valid packet is

$$r_m = \frac{1}{\rho_k}. \tag{4.14}$$

From this we can calculate the expected number of packets that will provide k innovative packets

$$\sum_{m=1}^{k} r_m = \sum_{m=1}^{k} \frac{1}{\rho_m}. \tag{4.15}$$

Figure 4.3 shows $\sum_{m=1}^{k} r_m - k$ the number of additional packets required to successfully construct a generator matrix. The results obtained by means of the formulae presented above are verified by the use of a Monte Carlo simulation. The Monte Carlo simulation generates independent and randomly encoded packets. As these packets are generated, they are collected in a set and the rank of each set is determined. The number of packets required to produce a set of full rank is measured and also depicted in Figure 4.3.

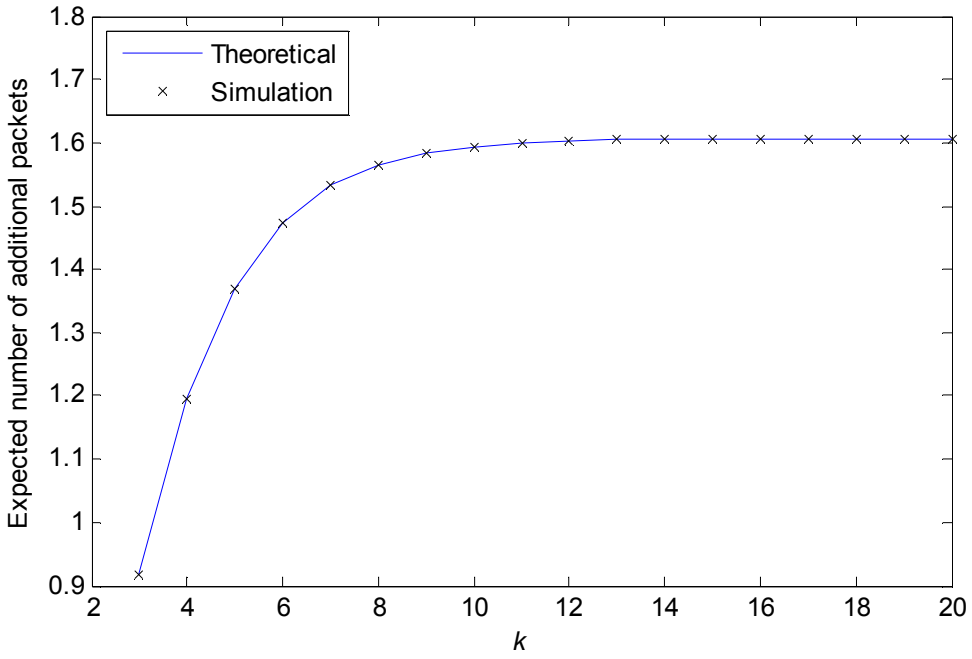

Figure 4.3: Expected number of packets to obtain k innovative packets

It can be seen that the number of extra packets required converges to approximately 1.6 for large k. This means that a receiver node would be able to obtain k linearly independent packets after approximately $N = k + 2$ collected packets. The trend seen in the above figure correlates with that of Figure 4.2, where the number of required packets increases as the probability of obtaining innovative packets declines. This corresponds with the results obtained in the literature [**69**].

4.4.3 Probability of $(n-k)$ additional packets being linearly independent and containing all source symbols

The above calculations show that there is a high probability that k linearly independent packets can be obtained after approximately $N = k + 2$ collected packets. To construct a valid $\boldsymbol{G_{IEC}}$ matrix, however, not only do we require k linearly independent packets, but also $(n-k)$ additional packets which have to be linearly independent of each other and contain all source symbols as well. In this section we calculate the probability of the remaining $(n-k)$ packets being linearly independent and containing all the source symbols.

The error correction capability of the linear error correction code relies on the structure of $\boldsymbol{G_{IEC}}$. Firstly, we calculate the probability of collecting sufficient row vectors to construct a generator matrix that corresponds with a linear code of $d_{min} \geq 2$, thus only being error detecting.

Firstly, we determine P_G the probability of obtaining a valid generator matrix $\boldsymbol{G_{IEC}}$ in the first n packets collected by a receiver node:

The probability $p_{n,k}$ of obtaining a full rank set (k linearly independent packets) within the first $N = n$ packets collected was determined in the previous section (4.11) to be

$$p_{n,k} = \prod_{i=0}^{k-1}\left(1 - \frac{1}{2^{n-i}}\right) \text{ for } n \geq k. \tag{4.16}$$

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

$$p_d = \prod_{i=1}^{n-k}\left(\frac{2^k - 2^{i-1}}{2^k - 1}\right) \tag{4.17}$$

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

$$p_s = 1 - \frac{\left[\binom{2^{k-1}-1}{n-k} \times k\right] - D}{\binom{2^k - 1}{n-k}}, \tag{4.18}$$

where

$$D = \sum_{i=3}^{k-2}(-1)^{k-i} \times \binom{2^i - 1}{n-k} \times \binom{k}{k-i}. \tag{4.19}$$

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

$$P_G = p_{n,k} \times p_d \times p_s \tag{4.20}$$

and is depicted in Figure 4.4. This figure also contains results from Monte Carlo simulations which were conducted to verify the obtained results. The simulation randomly and independently generates n packets and evaluates them to find whether there are two sets of linearly independent packets to construct a valid $\boldsymbol{G_{IEC}}$. The results obtained from the simulation match those of the mathematical model.

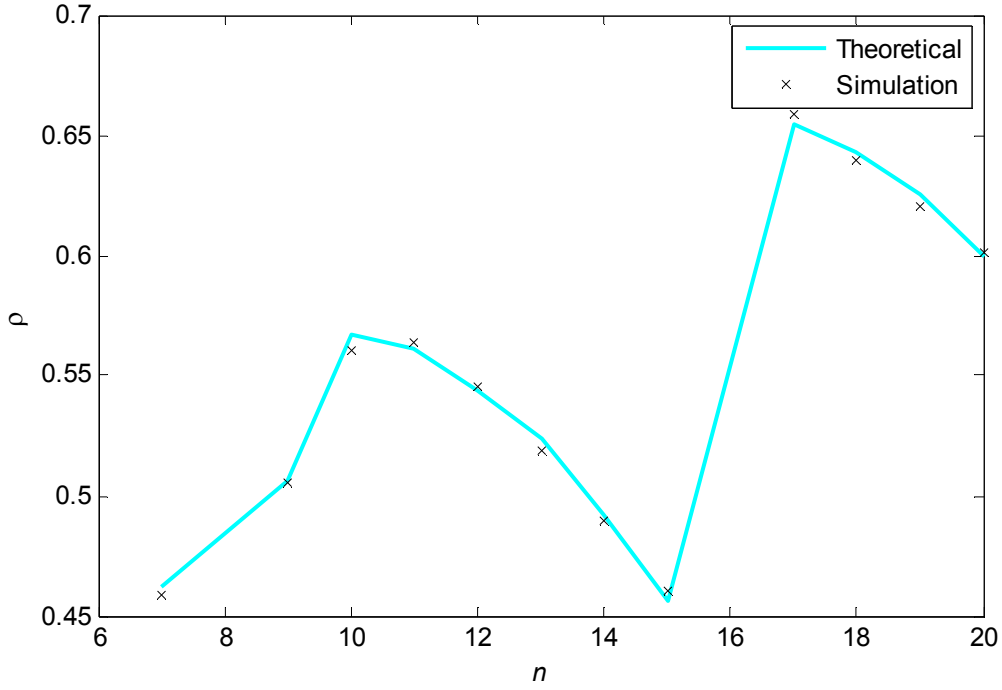

Figure 4.4: Probability of constructing a valid generator matrix after n packets received

This method constitutes an exhaustive evaluation of all the n packets received to form a valid $\boldsymbol{G_{IEC}}$. Although this method is computationally more expensive, the results show that an error can be detected with high probability after n received packets. A discussion on the results depicted in this figure is presented in Section 4.4.5.

4.4.4 Expected number of additional packets required to obtain a valid generator matrix

In this section we determine the expected number of packets required to guarantee the construction of a valid generator matrix, following the process described in Section 4.4.2.

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

$$Q_i = P_G \times (1 - P_G)^{i-1}, i = 1,2 \ldots. \tag{4.21}$$

The expectation of (4.21) is equal to

$$\sum_{i=1}^{\infty} i \times Q_i = \sum_{i=1}^{\infty} iP_G(1-P_G)^{i-1} = \frac{1}{P_G} \tag{4.22}$$

where P_G is shown in (4.20). The number of additional packets required to find the mth valid packet is shown in (4.14) from where we can calculate the expected number of packets that will provide n valid packets for the construction of a generator matrix

$$\sum_{m=1}^{n} r_m = \sum_{G=1}^{n} \frac{1}{P_G} \tag{4.23}$$

Figure 4.5 shows $\sum_{m=1}^{n} r_m - n$, the number of additional packets required to successfully construct a generator matrix as well as the results obtained via Monte Carlo simulations.

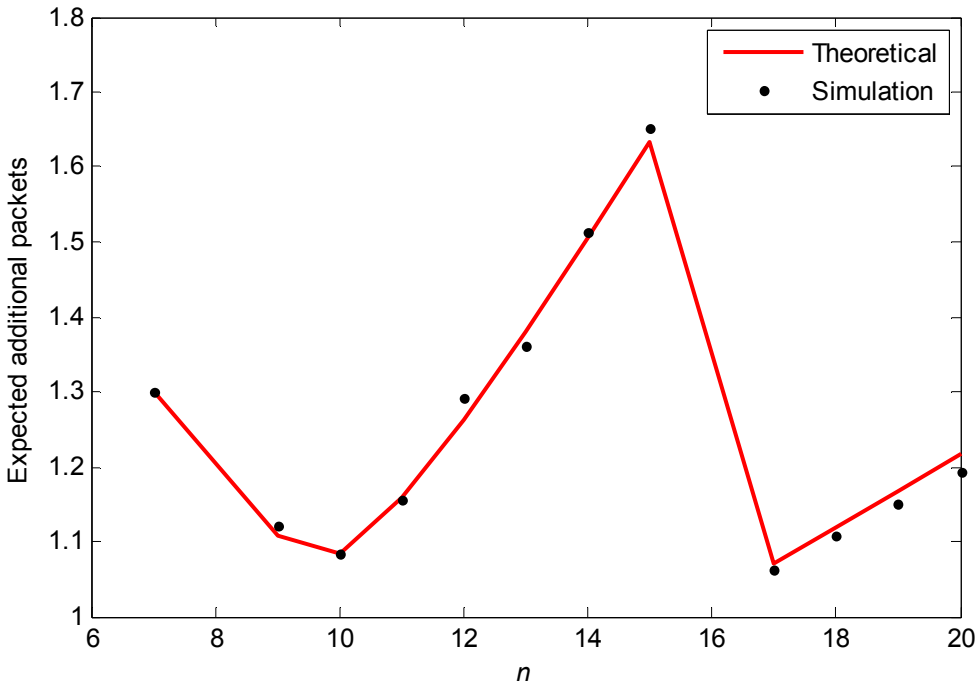

Figure 4.5: Expected number of additional packets required

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.

4.4.5 Discussion of obtained results

It can be seen in Figure 4.4 that the probability of constructing a valid $\boldsymbol{G_{IEC}}$ matrix for $n = 7$ and $n = 15$ dips to a minimum and thus maximises the number of additional packets required, shown in Figure 4.5. For block codes there is an important relationship between the block length n, dimension k and its error correcting capability, called the Hamming bound [**28**].

Definition 4-1: For any code $\mathcal{C} = (n, k, d)$ with $d \leq 2t + 1$, it exists that

$$|\mathcal{C}| \sum_{i=0}^{t} \binom{n}{i} \leq 2^n \tag{4.24}$$

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

This added redundancy is used by z to determine whether an error has occurred and whether it can be corrected [**28**]. For the purpose of implicit error correction the value of n is determined by the min-cut of the network, that is, 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 of (4.24), which means that 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 for perfect codes is lower and the number of expected additional packets higher. Hence the reason for $n = 15$ dipping to a minimum, thus maximising the number of additional packets required.

4.5 Simulation setup and results

In this section we aim to evaluate the redundancy required at receiver nodes to implement implicit error detection successfully. In order to do so, we try to find the correlation between the mathematical model developed in Section 4.4 and a practical RLNC network environment described in Chapter 2. 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 random 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.

4.5.1 Simulation setup

We base the experimental setup on that of [**5**], [**70**] 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 directed unit edges in $\mathcal{G}$ which represents the communication channels as described in Chapter 2. We consider a randomly generated network with $|\mathcal{V}| = 100$ nodes and a set of receivers $z \in Z$.

The data to be transmitted by the source node $s \in \mathcal{V}$ is modelled as k source symbols in the finite field $\mathbb{F}_q$. These source packets are multicast over the edges $e \in \mathcal{E}$ of network $\boldsymbol{\mathcal{G}}$. At each intermediate network node $v \subset \mathcal{V}$ the packets received on its incoming edges e are randomly and linearly combined over $\mathbb{F}_2$ to form a new encoded packet to be transmitted on the outgoing edge e. A global encoding vector of length k is included in the header of each outgoing packet. This describes the source packets linearly combined in the transmitted packet. A sink node z collects a set of $N \geq n$ encoded packets from the network where the global encoding vectors are stored as the column vectors of the $n \times k$ matrix $\boldsymbol{G_{IEC}}$.

4.5.2 Simulation methodology

Since we are interested in the effects of topology on the implicit error detection method, certain network parameters are chosen to remain constant throughout this simulation. Although these parameters may influence the implemented methods, they fall outside the scope of this thesis and will remain constant.

The following network parameters are specifically constrained for the purpose of this study:

1. *Network topology*. We assume that the nodes in the network simulation are not mobile nodes. When nodes are in fixed positions throughout an iteration, the min-cut of the network cannot be influenced. Also, during a single simulation set, no nodes enter or exit the network. As we aim to test the influence of network topology on the implicit error detection method, a static network topology is justified.

2. *Continuous transmission*. We assume that the source node continuously multicasts random linear combinations of source symbols. The transmission of packets from the source only stops once all the receiver nodes have successfully decoded the source data. The simulation is set up in this way in order to test the number of additional packets required to enable successful decoding at receiver nodes, not the efficiency of the network.

3. *Non-overlapping generations.* For this simulation we divided our source data into non-overlapping generations. With this setup, the possibility of constructing a valid generator matrix for a single set of k source symbols can be tested.

4. *Omission of payloads.* In these simulations tests are carried out only on the global encoding vectors of the received packets. Although a payload is present in practical network environments, these tests only evaluate the global encoding vectors of the packets. Therefore the payloads in these simulations are omitted to speed up the simulation process.

5. *Buffer sizes.* To ensure that the buffer size of the network nodes does not influence the results, each node has infinite in and out buffers.

We generate 200 random graphs for each simulation set. For each random graph, five instances are run with different seeds. This equates to 1000 simulation sets for a specified graph topology and value of k.

4.5.3 Network topology setup

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 [**70**].

The Érdos-Rényi Graph, ER(100, δ) = $(\mathcal{V}, \mathcal{E})$. This graph is constructed by randomly and independently including edges between all 100 nodes in the graph with probability δ. An example of this network topology is shown in Figure 4.6. Note that the example in Figure 4.6 only contains 20 nodes to visually illustrate the random and independent connections made between the nodes.

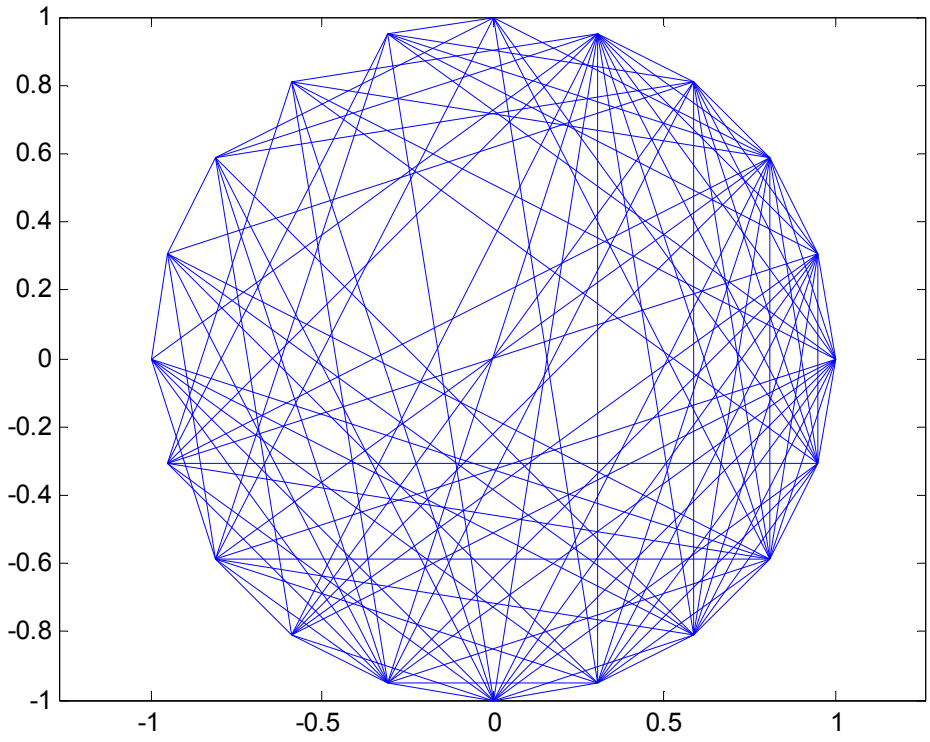

Figure 4.6: Example of an Érdos-Rényi graph for $|\mathcal{V}| = 20$

The Random Geometric Graph, RGG(100, l). This is formed by placing 100 nodes uniformly at random on a unit square with communication radius of l. Figure 4.7 gives an example of this network topology. Note that the example in Figure 4.7 only contains 30 nodes to visually illustrate the connections made between the nodes.

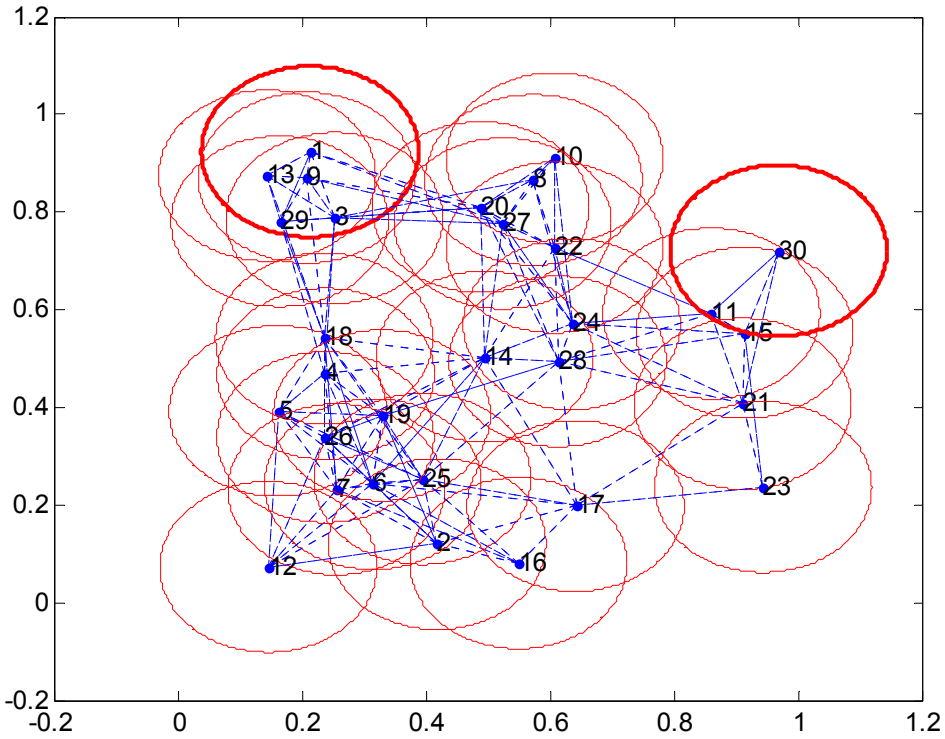

Figure 4.7: Example of a Random Geometric Graph for $|\mathcal{V}| = 30$

The values of δ in the ER graph and l for the RGG are specifically chosen so that the connectivity of the graphs is approximately the same to ensure min-cut $(s, z) \geq n$. This allows us to make a direct comparison between the two different network models.

4.5.4 Results

Error detection

We evaluated the number of additional packets required by the receiver nodes in order to construct a valid $\boldsymbol{G_{IEC}}$ that corresponds to a linear code of $d_{min} \geq 2$ where k packets are transmitted by the source. Figure 4.8 shows the number of additional packets required by the receiver nodes to successfully construct a valid generator matrix $\boldsymbol{G_{IEC}}$.

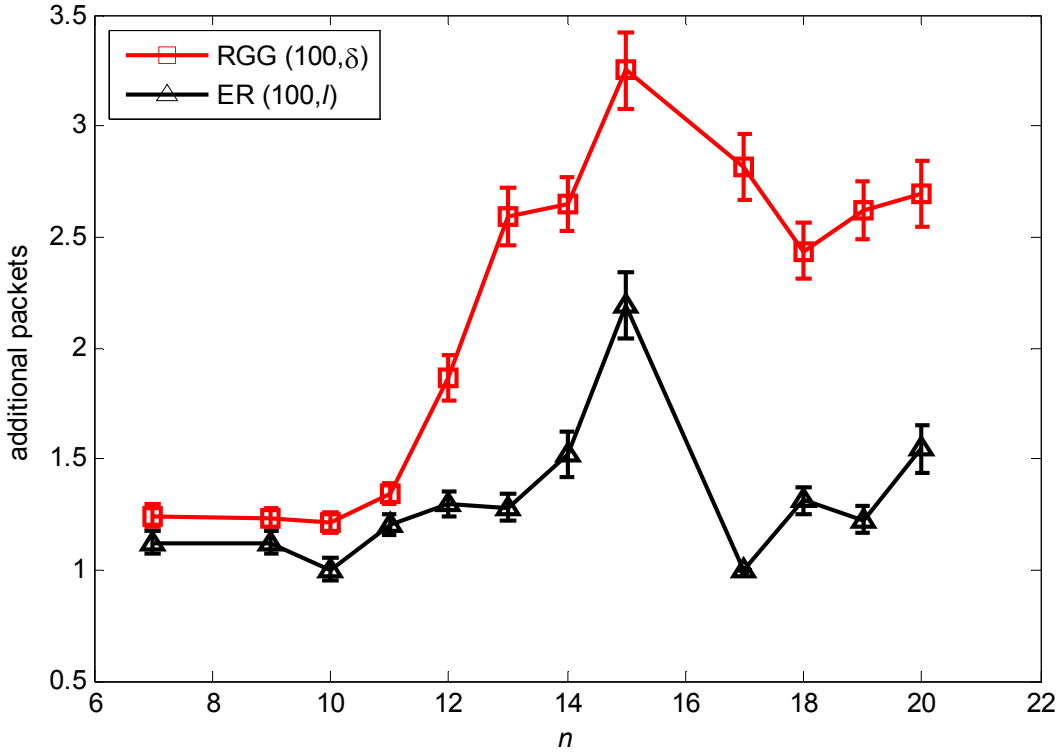

Figure 4.8: Number of additional packets required for error detection

From Figure 4.8 it can be seen that there is a significant difference between the results obtained for the RGG and the ER graphs.

The ER graph: When the results in Figure 4.8 are compared to the expected number of additional packets calculated in the analytical expression in Figure 4.5, 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 among 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.

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. Although each intermediate node randomly selects the packets that it uses for encoding, the new encoded packets are dependent on the packets present at the intermediate nodes, which in this case are not as well distributed as in the ER graphs. This results in more additional packets having to be received for successful error detection. This corresponds to the basic principles of RLNC [**5**], [**12**] as the lack of randomly and independently constructed packets causes linearly dependent packets with high probability.

Error correction

In this section, we determine the number of additional packets required to construct a $\boldsymbol{G_{IEC}}$ 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 $\boldsymbol{G_{IEC}}$ which encodes code words with Hamming distances $d_{min} \geq 3$.

In Chapter 3 the implementation of FEC as an outer error correction code is discussed. This requires a receiver node to obtain n packets with linearly independent global encoding vectors. In a network where coding is performed over $\mathbb{F}_2$, a receiver node can expect approximately two additional packets for the successful reception of sufficient packets to implement error correction, as can be seen in Figure 4.3 [**69**]. Thus, to implement an (n, k) error correction code in a practical RLNC network with the use of an outer code, a receiver node would have to collect approximately $(n + 2)$ packets from the network in order to successfully implement error correction.

We compare the two additional packets required for error correction when an outer code is implemented with the number of additional packets required when implicit error correction is implemented. For the implementation of implicit error correction, a valid generator matrix $\boldsymbol{G_{IEC}}$ that enables error correction must be constructed at the receiver node. The comparison of the additional packets required for the two methods is shown in Figure 4.9.

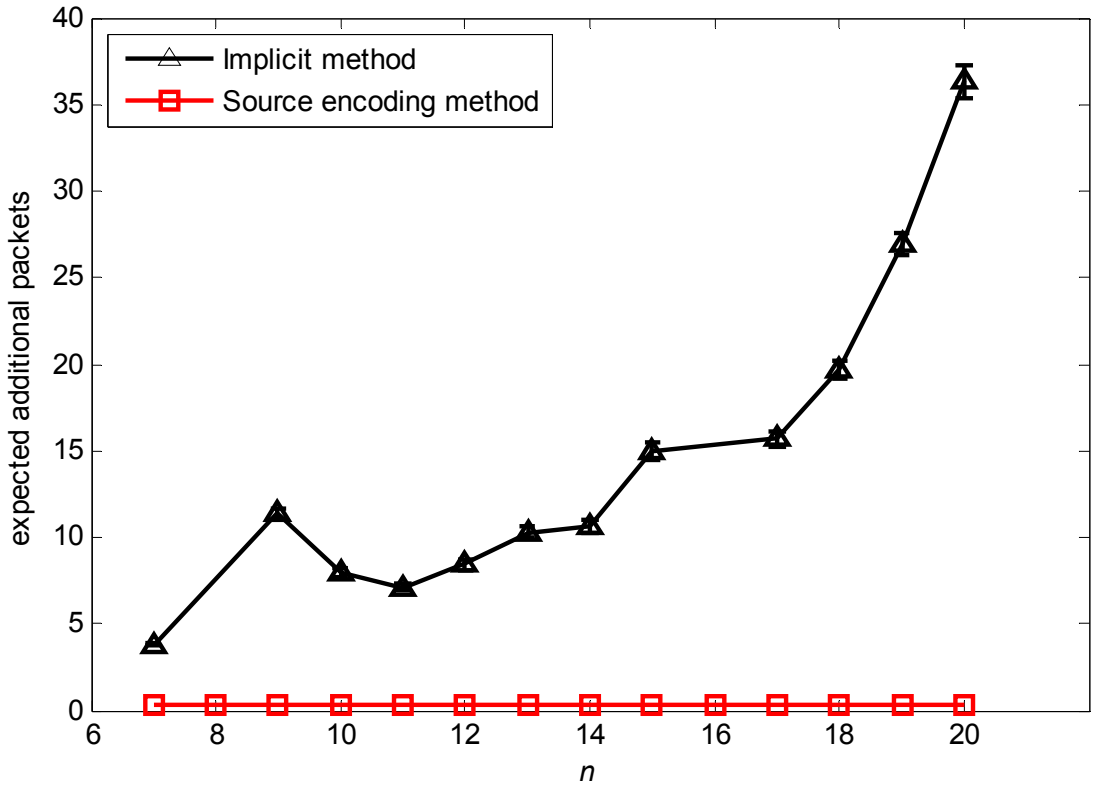

Figure 4.9: Number of extra packets required for $d_{min} = 3$ generator matrix.

It can be seen that the number of additional packets required for single error correction is very high and not practical. When an error correction code needs to be implemented in an RLNC network, the use of an outer error correction code at the source node would be far more effective than constructing a valid generator matrix $\boldsymbol{G_{IEC}}$ that enables error correction at the receiver nodes.

Discussion of obtained results

It was shown in Figure 4.3 that when encoding is performed over $\mathbb{F}_2$, a receiver node can expect approximately two additional packets for successful reception of k linearly independent packets. However, the work done in this chapter shows that with the addition of approximately three additional packets, a receiver node can implement single error detection without the addition of an outer code at the source node.

This method requires the transmission of k source symbols over the network, instead of n, which enables the receiver nodes to possibly detect a single error. The transmission of k source symbols into the network instead of n, can lead to intermediate nodes requiring less buffer space and performing fewer arithmetical operations during the random encoding of packets. The encoded packets also require a smaller overhead, as the number of source symbols is reduced.

4.6 Conclusion

In this chapter we improved and evaluated an implicit error detection technique from [**50**]. This method collects the packets implicitly formed by the RLNC process and constructs a generator matrix that can be used for error detection. We constructed an analytical expression that considers a network where the non-zero encoded packets received from the network by the receiver nodes are Gaussian distributed. This model was used to

- analyse the probability of constructing a valid $k \times n$ generator matrix after n received packets, and
- calculate the number of additional packets required to construct the valid generator matrix necessary to guarantee error detection in an RLNC network.

The analytical expression showed that the reception of approximately two additional packets can enable receiver nodes to detect a single packet error with high probability. This 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 ER 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 RGG is higher than shown by the analytical expression. The analytical expression that was developed describes the ER graph accurately, because packets are more randomly distributed. The RGG is not accurately described by a network which receives randomly distributed encoded packets, as the connections in the RGG network are more local and the packets obtained by a receiver node do not form a Gaussian distribution.

The analytical expression showed that this method is capable of detecting a single error with high probability when two additional packets are received. In the event where multiple packet errors occur, the receiver nodes would most probably be unable to detect it. This is due to the Hamming distance of the implemented code being $d_{min} = 2$ in most cases, only enabling the detection of a single error.

The implicit error detection method presented in this chapter is advantageous for use in networks with a large min-cut where the size of transmissions must be kept as small as possible due to limited resources. The source packets transmitted over the network with the implicit error detection method are shorter than packets where FEC codes are implemented as an outer code at the source. This leads to the use of less buffer space at the intermediate nodes, shorter switching time and lower computational complexity.

The reduction in buffer sizes of intermediate nodes and smaller overhead in packets lead to a more favourable environment for RLNC [**71**]. A study performed by [**71**] shows that network coding opportunities in a wireless network are more favourable when the transmitted packets are smaller. Wireless network scenarios exist where information packets are small and may only consist of a few bits [**72**], [**73**]. Accordingly, such networks can gain from this implicit error detection method as no additional data is added to the network, but error detection may be possible.

This single error detection method is not optimal, but does not effectively cost the network any additional resources when implemented. It shows that the redundancy generated by intermediate nodes in an RLNC network can not only be used for erasure correction, but possibly for error detection as well.