Codes for Graph Erasures

Lev Yohananov, *Student Member, IEEE*, and Eitan Yaakobi, *Senior Member, IEEE*

***Abstract*— Motivated by systems, where the information is represented by a graph such as neural networks, associative memories, and distributed systems. In this paper, we present a new class of codes, called *codes over graphs*. Under this paradigm, the information is stored on the edges of undirected or directed complete graphs, and a code over graphs is a set of graphs. A *node failure* is the event, where all edges in the neighborhood of the erased node have been erased. We say that a code over graphs can tolerate ρ node failures, if it can correct the erased edges of any ρ failed nodes in the graph. While the construction of optimal codes over graphs can be easily accomplished by MDS codes, their field size has to be at least $\mathcal{O}(n^2)$, when n is the number of nodes in the graph. In this paper, we present several constructions of codes over graphs with smaller field size. To accomplish this task, we use constructions of product codes and rank metric codes. Furthermore, we present optimal codes over graphs correcting two node failures over the binary field, when the number of nodes in the graph is a prime number. Last, we also provide upper bound on the number of nodes for optimal codes.**

***Index Terms*— Array codes, codes over graphs, EVENODD codes, product codes, rank metric codes.**

I. Introduction

The traditional setup to represent information is by a vector over some fixed alphabet. Although this commonly used model is the most practical one, especially for storage and communication applications, it does not necessarily fit all information systems. In this work we study a different approach where the information is represented by a *graph*. This model is motivated by several information systems. For example, in *neural networks*, the neural units are connected via *links* which store and transmit information between the neural units [7]. Similarly, in associative memories, the information is stored by associations between different data items [20]. These two examples mimic the brain functionality which stores and processes information by associations between the information units. Furthermore, representing information in a graph can model a distributed storage systems [4] while every two nodes can share a link with the information that is stored between the nodes. For example, a node may correspond to a user and the edges are files which are shared between two users, while self loops are the user's files.

A set of graphs called, *code over graphs*, is a new class of codes storing the information on the edges of the graph. In other words, each codeword of such a code is a graph with n *nodes* (vertices) and each edge stores a symbol over some fixed alphabet. There are two families of such codes. The first family consists of undirected complete graphs with self loops. In this case, the information is stored on each of the $\binom{n+1}{2}$ edges. Similarly, the second family consists of directed complete graphs with self loops and the information is stored on the n^2 edges of the graph. A *node failure* is the event where all the edges in the node's neighborhood have been erased, and the goal of this work is to construct codes over graphs that can efficiently correct node failures. Namely, we say that a code over graphs can correct ρ node failures if it is possible to correct the erased edges in the neighborhoods of any ρ failed nodes. We study node failures since they correspond to the events of failing neural units in a neural network, data loss in an associative memory, and unavailable or failed nodes in distributed storage systems. In case every node corresponds to a user, a node failure implies that the user's files as well as the ones that are shared with the user are erased. Furthermore, the information stored in a complete graph can be represented by an $n \times n$ array and a failure of the ith node corresponds to the erasure of the ith row and ith column in the array. Hence, this problem is translated to the problem of correcting *symmetric crisscross erasures* in square symmetric or non-symmetric arrays [14], [16], [17].

Assume a code over undirected graphs with n nodes such that every edge stores a symbol. If ρ nodes have failed then the number of edges that were erased is

$$\binom{n+1}{2} - \binom{n-\rho+1}{2} = n\rho - \binom{\rho}{2}. \quad (1)$$

Therefore, according to the Singleton bound, the number of *redundancy edges* for every code which tolerates ρ node failures is at least $n\rho - \binom{\rho}{2}$. Similarly, for the directed case, the failure of any ρ nodes translated to $2n\rho - \rho^2$ failed edges in the graph and thus the minimum number of redundancy edges of such a code is at least

$$n^2 - (n-\rho)^2 = 2n\rho - \rho^2. \quad (2)$$

A code over undirected, directed graphs which meets the lower bound (1), (2) on the number of redundancy edges, respectively, will be called an *optimal code over graphs*. While the construction of optimal codes meeting these bounds can be

Manuscript received June 23, 2018; revised January 31, 2019; accepted April 1, 2019. Date of publication April 11, 2019; date of current version August 16, 2019. This work was supported in part by the Israel Science Foundation under Grant 1624/14. This paper was presented in part at the IEEE International Symposium on Information Theory [23] and in part at the IEEE Information Theory Workshop [24].

The authors are with the Department of Computer Science, Technion—Israel Institute of Technology, Haifa 32000, Israel (e-mail: levyohananov@cs.technion.ac.il; yaakobi@cs.technion.ac.il).

Communicated by A. Jiang, Associate Editor for Coding Theory.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2019.2910040

easily accomplished by MDS codes, their field size has to be at least $\mathcal{O}(n^2)$. Our main goal in this work is the construction of codes over graphs with smaller fields.

Since every graph can be represented by its adjacency matrix, a natural approach to construct codes over graphs is by their adjacency matrices. Thus, this class of codes is quite similar to the class of *array codes*, such as maximum-rank array codes [14], d-codes [16], [17], B-codes [19], EVENODD codes [2], STAR codes [8], RDP code [9], X-codes [10], and regenerating codes [6], [12], [13], [15], [18], [21], [22]. However, there are several differences between classical array codes and codes over graphs. First, the adjacency matrix of a graph is a square matrix. Second, when the graphs are undirected, the adjacency matrices are symmetric. Third, a failure of the ith node in the graph corresponds to the failure of the ith row and the ith column in the adjacency matrix.

Most existing constructions of array codes are not designed for symmetric arrays, and they do not support this special row–column failure model. However, it is still possible to use existing code constructions and modify them to the special structure of the above erasure model in graphs. There are several candidates for this approach, such as product codes [1], [5], rank-metric codes [14], [16], [17], and variants of EVENODD codes [2].

The rest of this paper is organized as follows. In Section II, we formally define the graph models studied in this paper and some preliminary results. In Section III, we present codes over graphs correcting arbitrary number of node failures over a field of size at least $n-1$. In Section IV, we present binary non-optimal constructions with respect to the bound in (3) and (4). Our main result in the paper, presented in Section V, is an optimal binary code over undirected graphs correcting two node failures, when the number of nodes is prime. Then, in Section VI we show how to extend this construction for directed graphs. Lastly, in Section VII, we study bounds on the existence of optimal codes over graphs correcting ρ node failures. Section VIII concludes the paper.

II. Definitions and Preliminaries

In this section we formally define the tools and the definitions used throughout the paper. For a positive integer n, the set $\{0, 1, \ldots, n-1\}$ will be denoted by $[n]$. For a prime power q, $\mathbb{F}_q$ is a finite field of size q. A linear code of length n and dimension k over $\mathbb{F}_q$ will be denoted by $[n, k]_q$ or $[n, k, d]_q$, where d denotes its minimum distance.

We will denote a graph by $G = (V_n, E)$, where $V_n = \{v_0, v_1, \ldots, v_{n-1}\}$ is the set of n nodes (vertices) and $E \subseteq V_n \times V_n$ is its edge set. In this paper, we only study complete graphs with self loops that can be directed or undirected. The edge set of a directed graph G over Σ will be defined by $E = V_n \times V_n$, with a labeling function $L_{\mathcal{D}} : V_n \times V_n \to \Sigma$. We will use the notation $G = (V_n, L_{\mathcal{D}})$ for such graphs, since we can fully characterize the graph G by its vertex set V_n and its labeling function $L_{\mathcal{D}}$. Similarly, the edge set of an undirected graph G over Σ will be defined by $E = \{(v_i, v_j) \mid (v_i, v_j) \in V_n \times V_n, i \geqslant j\}$, with a labeling function $L_{\mathcal{U}} : V_n \times V_n \to \Sigma$ and we will use the notation $G = (V_n, L_{\mathcal{U}})$ for such graphs. By a slight abuse of notation, every undirected edge in the graph will be denoted by $\langle v_i, v_j \rangle$ where the order in this pair does not matter, that is, the notation $\langle v_i, v_j \rangle$ is identical to the notation $\langle v_j, v_i \rangle$. Note that a directed, undirected graph with n vertices has n^2, $\binom{n+1}{2}$ edges, respectively. A graph $G = (V_n, L)$ is a general definition that refers to both directed and undirected graphs, while it will be clear from the context which case the notation refers to.

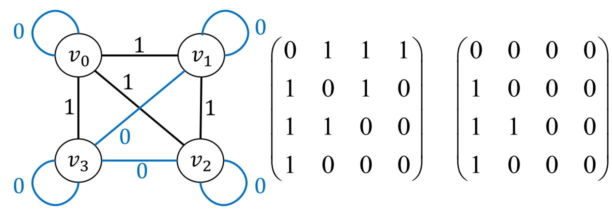


Fig. 1. The undirected graph G, its adjacency matrix A_G, and its lower-triangle-adjacency matrix A'_G.

The ***adjacency matrix*** of a graph $G = (V_n, L)$ is an $n \times n$ matrix over Σ denoted by $A_G = [a_{i,j}]_{i=0,j=0}^{n-1,n-1}$, where $a_{i,j} = L(v_i, v_j)$ for a directed graph, and $a_{i,j} = L\langle v_i, v_j \rangle$ for an undirected graph, while $i, j \in [n]$. Notice that the adjacency matrix of an undirected graph is symmetric. For undirected graphs, we also define the ***lower-triangle-adjacency matrix*** of G to be the $n \times n$ matrix $A'_G = [a'_{i,j}]_{i=0,j=0}^{n-1,n-1}$ such that $a'_{i,j} = a_{i,j}$ if $i \geqslant j$ and otherwise $a'_{i,j} = 0$. The ***upper-triangle-adjacency matrix*** is defined similarly. We define the ***zero graph*** by $G_{\mathbf{0}}$ if for all $i, j \in [n]$, we have $a_{i,j} = 0$.

The next example demonstrates the above definitions for undirected graphs.

Example 1. Let G be a complete undirected graph with self loops over $\mathbb{F}_2$ and let $V_4 = \{v_0, v_1, v_2, v_3\}$ be its node set. The graph G, its adjacency matrix A_G, and lower-triangle-adjacency matrix A'_G are shown in Fig. 1, where the edges $\langle v_0, v_1 \rangle$, $\langle v_0, v_2 \rangle$, $\langle v_0, v_3 \rangle$ and $\langle v_1, v_2 \rangle$ are labeled with 1 and the rest of the edges are labeled with 0.

Let Σ be a ring and G_1 and G_2 be two graphs over Σ with the same node set V. The operator "+" between G_1 and G_2 over Σ, is defined by $G_1 + G_2 = G_3$, where G_3 is the unique graph satisfying $A_{G_1} + A_{G_2} = A_{G_3}$. Similarly, the operator "$\cdot$" between G_1 and an element $\alpha \in \Sigma$, is denoted by $\alpha \cdot G_1 = G_3$, where G_3 is the unique graph satisfying $\alpha \cdot A_{G_1} = A_{G_3}$.

Definition 1. *Let V_n be the set of nodes $V_n = \{v_0, \ldots, v_{n-1}\}$. A **code over directed graphs** over Σ of length n and size M is a set of directed graphs $\mathcal{C}_{\mathcal{D}} = \{G_i = (V_n, L_{\mathcal{D}_i}) | i \in [M]\}$ over Σ, denoted by $\mathcal{D}$-$(n, M)_\Sigma$. Similarly a **code over undirected graphs** over Σ of length n and size M is a set of undirected graphs $\mathcal{C}_{\mathcal{U}} = \{G_i = (V_n, L_{\mathcal{U}_i}) | i \in [M]\}$ over Σ, denoted by $\mathcal{U}$-$(n, M)_\Sigma$. In case that $\Sigma = \{0, 1\}$, the directed and the undirected codes over graphs will simply be denoted by $\mathcal{D}$-(n, M) and $\mathcal{U}$-(n, M). A **code over graphs** $\mathcal{C}_{\mathcal{G}}$ is a general definition that refers to both codes over directed graphs and codes over undirected graphs, while the meaning will be clear from the context.*

*The **dimension** of a code over directed, undirected graphs $\mathcal{C}_\mathcal{D}$, $\mathcal{C}_\mathcal{U}$ is $k_\mathcal{D} = \log_{|\Sigma|} M$, $k_\mathcal{U} = \log_{|\Sigma|} M$, respectively. The **rate** of a code over directed, undirected graphs is $R_\mathcal{D} = k_\mathcal{D}/n^2$, $R_\mathcal{U} = k_\mathcal{U}/\binom{n+1}{2}$ and the **redundancy** is defined to be $r_\mathcal{D} = n^2 - k_\mathcal{D}$, $r_\mathcal{U} = \binom{n+1}{2} - k_\mathcal{U}$, respectively.*

*A code over directed graphs $\mathcal{C}_\mathcal{D}$ over a ring Σ will be called **linear** if for every $G_1, G_2 \in \mathcal{C}_\mathcal{D}$ and $\alpha, \beta \in \Sigma$ it holds that $\alpha G_1 + \beta G_2 \in \mathcal{C}_\mathcal{D}$. A linear code over undirected graphs is defined similarly. We denote such codes over directed, undirected graphs by $\mathcal{D}$-$[n, k_\mathcal{D}]_\Sigma$, $\mathcal{U}$-$[n, k_\mathcal{U}]_\Sigma$, respectively.*

*A linear code over directed, undirected graphs will be called **systematic** if the first k nodes contain the k^2, $\binom{k+1}{2}$ unmodified information symbols on their edges, respectively. All other $n^2 - k^2$, $\binom{n+1}{2} - \binom{k+1}{2}$ edges in the graph are called **redundancy edges**, respectively. In this case we say that there are k **information nodes** and $r = n - k$, **redundancy nodes**. The number of **information edges** is $k_\mathcal{D} = k^2$, $k_\mathcal{U} = \binom{k+1}{2}$, the redundancy is $r_\mathcal{D} = n^2 - k^2$, $r_\mathcal{U} = \binom{n+1}{2} - \binom{k+1}{2}$, and the rate is $R_\mathcal{D} = k^2/n^2$, $R_\mathcal{U} = \binom{k+1}{2}/\binom{n+1}{2}$ for directed, undirected codes over graphs, respectively. We denote such a code by $\mathcal{SD}$-$[n, k]_\Sigma$ for directed codes over graphs and $\mathcal{SU}$-$[n, k]_\Sigma$ for undirected codes over graphs.*

Definition 2. *Let $G = (V_n, L_\mathcal{D})$ be a directed graph. For $i \in [n]$, the **out-neighborhood edge set**, **in-neighborhood edge set**, of the i-th node is defined to be the set*

$$N_i^{\text{out}} = \{(v_i, v_j) \mid j \in [n]\}, \; N_i^{\text{in}} = \{(v_j, v_i) \mid j \in [n]\},$$

*respectively, and the **neighborhood edge set** of the i-th node is the set $N_i = N_i^{\text{out}} \cup N_i^{\text{in}}$. Note that the i-th out-neighborhood, in-neighborhood edge set, corresponds to the i-th row, column, in an adjacency matrix A_G, respectively, and the i-th neighborhood edge set is the union of the i-th column and the i-th row in the adjacency matrix. Similarly, the neighborhood edge set of the i-th node of an undirected graph $G = (V_n, L_\mathcal{U})$ is defined by $N_i = \{\langle v_i, v_j \rangle \mid j \in [n]\}$, which corresponds to the i-th column and row in an adjacency matrix A_G.*

*The **node failure** of the i-th node is the event in which all the edges in the neighborhood set of the i-th node, i.e. N_i, are erased. We will also denote this edge set by F_i and refer to it by the **failure set** of the i-th node. For convenience, for directed graphs we also define the **out-failure set**, **in-failure set** of the i-th node by $F_i^{\text{out}} = N_i^{\text{out}}$, $F_i^{\text{in}} = N_i^{\text{in}}$, respectively.*

*A code over directed graphs is called a **directed ρ-node-erasure-correcting code** if it can correct any failure of at most ρ nodes in each of its graphs. An **undirected ρ-node-erasure-correcting code** is defined similarly. A **ρ-node-erasure-correcting code** is a general definition that refers to both codes over directed and undirected graphs.*

According to the Singleton bound, we deduce that the minimum redundancy $r_\mathcal{D}$, $r_\mathcal{U}$ of any directed, undirected ρ-node-erasure-correcting code of length n, satisfies

$$r_\mathcal{D} \geqslant n^2 - (n-\rho)^2 = 2n\rho - \rho^2, \quad (3)$$

$$r_\mathcal{U} \geqslant \binom{n+1}{2} - \binom{n-\rho+1}{2} = n\rho - \binom{\rho}{2}, \quad (4)$$

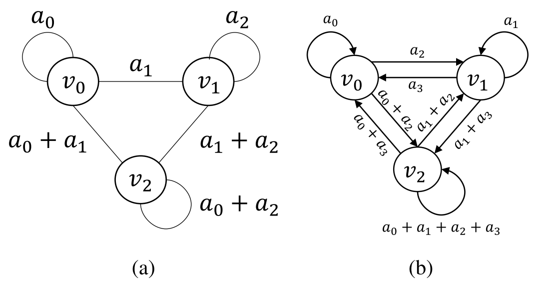


Fig. 2. Two constructions of optimal binary single-node-erasure-correcting codes $\mathcal{SG}$-[3, 2]. (a) Undirected case. (b) Directed case.

respectively. A code over directed, undirected graphs satisfying the first, second inequality with equality will be called ***optimal***, respectively. In this paper we study only linear codes. We also state that every optimal code according to the bound (3) or (4) is systematic as it was defined above. Hence, for systematic codes over graphs the number of redundancy nodes is at least ρ. Note that for all n and ρ, one can always construct an optimal directed ρ-node-erasure-correcting code from an $[n^2, (n-\rho)^2, 2n\rho - \rho^2 + 1]$ MDS code. Similarly, one can always construct an optimal undirected ρ-node-erasure-correcting code from an $[\binom{n+1}{2}, \binom{n-\rho+1}{2}, n\rho - \binom{\rho}{2} + 1]$ MDS code. However, in both cases the field size of the code over graphs will be at least $\Theta(n^2)$. Our goal in this work is to construct ρ-node-erasure-correcting codes over smaller fields. When possible, we seek the field size to be binary and in any event at most $\mathcal{O}(n)$.

The next example exemplifies the definitions of codes over undirected graphs.

Example 2. The following codes given in Fig. 2, are systematic binary single-node-erasure-correcting codes of length 3. The left figure illustrates an undirected code over graphs, and the right figure illustrates a directed code over graphs. Both constructions store the information on edges of the complete subgraph of nodes v_0 and v_1. For the undirected case, the neighborhood set of each node belongs to a simple parity code of length 3. Similarly for the directed case, the out-neighborhood set and the in-neighborhood set of each node belongs to a simple parity code of length 3.

The code construction from Example 2 is easily extended for arbitrary number of nodes. For undirected graphs this construction is formulated as follows.

Construction 1 Let $n \geqslant 2$ be a positive integer. The code over undirected graphs $\mathcal{C}_{\mathcal{U}_1}$ is defined as follows.

$$\mathcal{C}_{\mathcal{U}_1} = \{G = (V_n, L_\mathcal{U}) \; : \; \forall i \in [n], \; \sum_{j=0}^{n-1} L_\mathcal{U}\langle v_i, v_j \rangle = 0\}.$$

As mentioned above, the constraints imposed in this constructions state that the edges in the neighborhood of each vertex in the graph form a simple parity code, so we call it a *neighborhood constraint*. In the adjacency matrix, that means that each row and column belongs to a simple parity code. The correctness of this construction is proved in the following theorem.

Theorem 3. *The code $\mathcal{C}_{\mathcal{U}_1}$ is an optimal binary undirected single-node-erasure-correcting code $\mathcal{SU}$-$[n, n-1]$.*

Proof: Suppose that the node v_i is erased. Then, for each node v_s such that $v_s \neq v_i$, the edge $\langle v_i, v_s\rangle$ is corrected by

$$L_{\mathcal{U}}\langle v_i, v_s\rangle = \sum_{j=0, j\neq i}^{n-1} L_{\mathcal{U}}\langle v_s, v_j\rangle,$$

and the only uncorrected edge left is the self loop $\langle v_i, v_i\rangle$. Therefore, the edge $\langle v_i, v_i\rangle$ is corrected by

$$L_{\mathcal{U}}\langle v_i, v_i\rangle = \sum_{j=0, j\neq i}^{n-1} L_{\mathcal{U}}\langle v_i, v_j\rangle.$$

Notice that the code $\mathcal{C}_{\mathcal{U}_1}$ is of size $\binom{n}{2}$, so its redundancy satisfies the bound in (4) with equality and thus the code is optimal. ■

The construction of an optimal binary directed single-node-erasure-correcting code $\mathcal{SD}$-$[n, n-1]$ is similar.

Next we define a distance metric over graphs that will be used in the construction of codes correcting node failures.

Definition 4. *Let $G = (V_n, L)$ be a graph and let E be the set of all nonzero labeled edges of G, i.e., $E = \{e \in V_n \times V_n \mid L(e) \neq 0\}$. A **vertex cover** W of G is a subset of V_n such that for each $(v_i, v_j) \in E$ (or $\langle v_i, v_j\rangle \in E$ in an undirected case) either $v_i \in W$ or $v_j \in W$. Then, the **graph weight** of G is defined by*

$$w_{\mathcal{G}}(G) = \min_{W \text{ is a vertex cover of } G}\{|W|\}.$$

*Intuitively, the value $w_{\mathcal{G}}(G_1 - G_2)$ is simply the minimum number of nodes whose removals make G_1 and G_2 identical. The **graph distance** between two graphs G_1, G_2 will be denoted by $d_{\mathcal{G}}(G_1, G_2)$ and it holds that $d_{\mathcal{G}}(G_1, G_2) = w_{\mathcal{G}}(G_1 - G_2)$.*

Lemma 5. *The graph distance is a metric.*

Proof:

1) Clearly, $d_{\mathcal{G}}(G_1, G_2) \geqslant 0$ since the vertex cover is defined to be non-negative, and by definition of the graph weight $d_{\mathcal{G}}(G_1, G_2) = 0$ if and only if $G_1 = G_2$.
2) Symmetry: $d_{\mathcal{G}}(G_1, G_2) = w_{\mathcal{G}}(G_1 - G_2) = w_{\mathcal{G}}(G_2 - G_1) = d_{\mathcal{G}}(G_2, G_1)$, since each edge of the graph $G = (G_1 - G_2)$ has a non-zero label if and only if it has a non-zero label in the graph $G' = (G_2 - G_1)$.
3) The triangle inequality:

$$\begin{aligned} d_{\mathcal{G}}(G_1, G_2) &= w_{\mathcal{G}}(G_1 - G_2) = w_{\mathcal{G}}(G_1 - G_3 + G_3 - G_2) \\ &= \min_{W \text{ v.c. of } (G_1 - G_3) + (G_3 - G_2)}\{|W|\} \\ &\leqslant \min_{W \text{ v.c. of } G_1 - G_3}\{|W|\} + \min_{W \text{ v.c. of } G_3 - G_2}\{|W|\} \\ &= w_{\mathcal{G}}(G_1 - G_3) + w_{\mathcal{G}}(G_3 - G_2) \\ &= d_{\mathcal{G}}(G_1, G_3) + d_{\mathcal{G}}(G_3, G_2), \end{aligned}$$

where the inequality holds since each non-zero labeled edge of the graph $G = ((G_1 - G_3) + (G_3 - G_2))$ is a non-zero labeled edge of either the graph $G' = (G_1 - G_3)$ or $G'' = (G_3 - G_2)$.

■

The ***minimum distance*** of a code over graphs $\mathcal{C}_{\mathcal{G}}$, denoted by $d(\mathcal{C}_{\mathcal{G}})$, is a minimum graph distance between any two distinct graphs in $\mathcal{C}_{\mathcal{G}}$, that is

$$d(\mathcal{C}_{\mathcal{G}}) = \min_{G_1 \neq G_2 \; G_1, G_2 \in \mathcal{C}_{\mathcal{G}}}\{d_{\mathcal{G}}(G_1, G_2)\},$$

and in case the code is linear

$$d(\mathcal{C}_{\mathcal{G}}) = \min_{G \in \mathcal{C}_{\mathcal{G}}, G \neq G_{\mathbf{0}}}\{w_{\mathcal{G}}(G)\}.$$

Theorem 6. *A linear code over graphs $\mathcal{C}_{\mathcal{G}}$ is a ρ-node-erasure-correcting code if and only if its minimum distance satisfies $d(\mathcal{C}_{\mathcal{G}}) \geqslant \rho + 1$.*

Proof: Assume that the minimum distance of $\mathcal{C}_{\mathcal{G}}$ satisfies $d(\mathcal{C}_{\mathcal{G}}) \geqslant \rho + 1$. Let Z be a received graph after having ρ node failures of a graph $G \in \mathcal{C}_{\mathcal{G}}$. Let $\mathcal{D}$ be a decoder of $\mathcal{C}_{\mathcal{G}}$, $\mathcal{D}(Z) \in \mathcal{C}_{\mathcal{G}}$ such that $\mathcal{D}(Z)$ is consistent with Z on all non-erased edges. We will show that $\mathcal{D}(Z) = G$ is the unique solution tolerating ρ node failures. Assume that there are two different graphs $G_1, G_2 \in \mathcal{C}_{\mathcal{G}}$ that are consistent with Z on all the non-erased edges. Therefore, we have $w_{\mathcal{G}}(G_1 - G_2) = d_{\mathcal{G}}(G_1, G_2) \leqslant \rho \leqslant d(\mathcal{C}_{\mathcal{G}}) - 1$, since all of the edges of the graph $G' = (G_1 - G_2)$ can be covered only by the failed nodes. Thus, we get a contradiction.

Next, suppose that $\mathcal{C}_{\mathcal{G}}$ has a minimum distance less than $\rho + 1$. We will show that there is no decoder that can correct any ρ node failures. Assume that there are two distinct graphs $G_1, G_2 \in \mathcal{C}_{\mathcal{G}}$ such that $d_{\mathcal{G}}(G_1, G_2) \leqslant d(\mathcal{C}_{\mathcal{G}}) - 1 < \rho$. Denote by G the graph $G_1 - G_2$, where $w_{\mathcal{G}}(G) = w_{\mathcal{G}}(G_1 - G_2) = d_{\mathcal{G}}(G_1, G_2)$, so $w_{\mathcal{G}}(G) < \rho$. Let W be a vertex cover of G and assume that the nodes of W in G_1 were erased. Since $G = (G_1 - G_2)$, the graphs G_1 and G_2 are consistent on all the non-erased edges of G_1, and the decoder of $\mathcal{C}_{\mathcal{G}}$ will not be able to correct such node failures. ■

III. Optimal Multiple-Node-Erasure-Correcting Codes Over Linear Field Size

In the previous section we saw that optimal ρ-node-erasure-correcting codes are easy to construct over a field of size $\Theta(n^2)$. In this section we will present two constructions that reduce the large field to $\Theta(n)$. Namely, we show an optimal construction of directed, undirected ρ-node-erasure-correcting codes $\mathcal{SD}$-$[n, n-\rho]_{\mathbb{F}_q}$, $\mathcal{SU}$-$[n, n-\rho]_{\mathbb{F}_q}$ for all n and ρ, where q is a prime power greater than $n-1$, respectively. This result is developed using the construction of product codes that were introduced by Elias in [5] and was discussed by Abramson in [1].

A. Optimal Codes Over Directed Graphs

We first review the construction of product codes. Let $\mathcal{C}_1, \mathcal{C}_2$ be a linear code with parameters $[n_1, k_1, d_1]_q, [n_2, k_2, d_2]_q$, respectively. Denote by H_1, H_2 the parity-check matrix of $\mathcal{C}_1, \mathcal{C}_2$, respectively. Then, the ***product code*** of $\mathcal{C}_1, \mathcal{C}_2$, denoted by $\mathcal{P}(\mathcal{C}_1, \mathcal{C}_2)$, is defined by

$$\mathcal{P}(\mathcal{C}_1, \mathcal{C}_2) = \{A \in \mathbb{F}_q^{n_1 \times n_2} \mid H_1 A = A H_2^{\top} = 0\}.$$

It was shown in [11] that $\mathcal{P}(\mathcal{C}_1, \mathcal{C}_2)$ is an $[N, K, D]_q$ linear code with $N = n_1 n_2$, $K = k_1 k_2$ and $D = d_1 d_2$. Therefore, according to the definition of $\mathcal{P}(\mathcal{C}_1, \mathcal{C}_2)$, each column of $A \in \mathcal{P}(\mathcal{C}_1, \mathcal{C}_2)$ is a codeword of $\mathcal{C}_1$ and each row of $A \in \mathcal{P}(\mathcal{C}_1, \mathcal{C}_2)$ is a codeword of $\mathcal{C}_2$. In case where $\mathcal{C}_1 = \mathcal{C}_2 = \mathcal{C}$, the code is denoted by $\mathcal{P}(\mathcal{C})$. If $\mathcal{C}$ is an $[n, k, d]_q$ code, then $\mathcal{P}(\mathcal{C}) = \{A \in \mathbb{F}_q^{n\times n} \mid HA = AH^\top = 0\}$ is an $[N, K, D]$ linear code with $N = n^2$, $K = k^2$ and $D = d^2$, where each row and each column of $A \in \mathcal{P}(\mathcal{C})$ is a codeword of $\mathcal{C}$ [11].

We are now ready to present a construction of optimal ρ-node-erasure-correcting codes over directed graphs with a field size not smaller than $n-1$. Let us consider the adjacency matrix of each graph in the code in order to explain the main idea of the construction. Each row and column in the adjacency matrix belongs to an $[n, n-\rho, \rho+1]_q$ MDS code, which will be denoted by $\mathcal{C}$, where $q \geqslant n-1$. Equivalently, each in-neighborhood edge set and out-neighborhood edge set of every node v, $v \in V_n$ is a codeword in $\mathcal{C}$. This construction is formalized as follows.

Construction 2 Let n and ρ be two positive integers such that $\rho < n$. Let $\mathcal{C}$ be an $[n, n-\rho, \rho+1]_q$ MDS code, for $q \geqslant n-1$, and let $\mathcal{P}(\mathcal{C})$ be its product code. The code $\mathcal{C}_{\mathcal{D}_1}$ is defined as follows,

$$\mathcal{C}_{\mathcal{D}_1} = \{G = (V_n, L_{\mathcal{D}}) \mid A_G \in \mathcal{P}(\mathcal{C})\}.$$

The correctness of Construction 2 is proved in the next theorem.

Theorem 7. *For all ρ and n such that $\rho < n$, the code $\mathcal{C}_{\mathcal{D}_1}$ is an optimal directed ρ-node-erasure-correcting code $\mathcal{D}$-$[n, k_{\mathcal{D}} = (n-\rho)^2]_{\mathbb{F}_q}$, where $q \geqslant n-1$.*

Proof: As mentioned before, the dimension of $\mathcal{P}(\mathcal{C})$ is $(n-\rho)^2$, therefore $\mathcal{C}_{\mathcal{D}_1}$ is a linear code over graphs with dimension $k_{\mathcal{D}} = (n-\rho)^2$. Let $G = (V_n, L_{\mathcal{D}})$, $G \in \mathcal{C}_{\mathcal{D}_1}$ be a graph and assume that ρ of its nodes are erased, where their indices are denoted by the set $J \subseteq [n]$. Let $U = \{v_i \in V_n \mid i \in J\}$ and $W = V_n \setminus U$. The decoding of the erased edges is invoked in two steps. In the first step, all incoming and outgoing edges of each $w \in W$ are corrected, and in the second step the remaining incoming and outgoing edges of each $u \in U$ are corrected.

1) Since every row and every column of A_G is a codeword of $\mathcal{C}$, the rows and columns with at most ρ erasures can be corrected. For every $i \in [n] \setminus J$ the i-th row and column has exactly ρ erasures, and therefore the incoming and outgoing edges of each $w \in W$ are corrected.
2) For each $w \in W$ exactly one incoming edge and exactly one outgoing edge of each $u \in U$ was corrected in the first step. Since $|W| = n-\rho$, the number of uncorrected incoming and outgoing edges for each $u \in U$ is ρ. Therefore the incoming and outgoing neighborhoods of each $u \in U$ can be corrected by the decoder of $\mathcal{C}$ as well. ∎

Note that the code $\mathcal{C}_{\mathcal{D}_1}$ is also systematic where its first $n-\rho$ nodes are the information nodes. In the adjacency matrix, this corresponds to having the information symbols in the upper left $(n-\rho) \times (n-\rho)$ matrix. Then, each of the first $n-\rho$ columns encoded with the systematic MDS code $\mathcal{C}$, and then the same procedure is invoked on each of the n rows. In the next section, we will construct similar codes over undirected graphs.

B. Optimal Codes Over Undirected Graphs

The construction of optimal codes over undirected graphs can be established by taking a sub-code of the product code $\mathcal{P}(\mathcal{C})$ which consists of only symmetric matrices. Let $\mathcal{C}$ be a linear code with parameters $[n, k, d]_q$. Denote by H the parity-check matrix of $\mathcal{C}$ and denote by G a generator matrix of $\mathcal{C}$. Then, the ***symmetric product code*** of $\mathcal{C}$, denoted by $\mathcal{H}(\mathcal{C})$, is defined by

$$\mathcal{H}(\mathcal{C}) = \{A \in \mathbb{F}_q^{n\times n} \mid HA = 0, A = A^\top\}.$$

First we will show that $\mathcal{H}(\mathcal{C})$ is an $[N, K, D]$ linear code with $N = n^2$, $K = \binom{k+1}{2}$ and $D = d^2$.

Lemma 8. *The symmetric product code $\mathcal{H}(\mathcal{C})$ is an $[N, K, D]$ linear code with $N = n^2$, $K = \binom{k+1}{2}$ and $D = d^2$.*

Proof: The code $\mathcal{H}(\mathcal{C})$ is linear since it is defined by parity-check equations. Since $\mathcal{H}(\mathcal{C})$ is a product code, its minimum distance is $D = d^2$. Let $u_0, u_1, \ldots, u_{\binom{k+1}{2}-1}$ be information symbols over $\mathbb{F}_q$ that will be stored in a symmetric matrix U that will be called an *information matrix*. Each information matrix $U \in \mathbb{F}_q^{k\times k}$, will be encoded by $A = G^\top UG$, and it is straightforward to verify that $HA = AH^\top = 0$. We show that if $U = U^\top$ then,

$$A = G^\top UG = G^\top U^\top G = (UG)^\top (G^\top)^\top = (G^\top UG)^\top = A^\top.$$

Moreover, if $A = A^\top$ then

$$\begin{aligned} A &= A^\top \\ G^\top UG &= (G^\top UG)^\top \\ G^\top UG &= (UG)^\top (G^\top)^\top \\ G^\top UG &= G^\top U^\top G, \end{aligned}$$

and since G is a full row rank matrix, it implies that $U = U^\top$. Therefore, the dimension of $\mathcal{H}(\mathcal{C})$ is equal to the dimension of $\{U \in \mathbb{F}_q^{k\times k} \mid U = U^\top\}$, that is $\binom{k+1}{2}$. ∎

This construction for undirected graphs is formalized as follows.

Construction 3 Let n and ρ be two positive integers such that $n > \rho$. Let $\mathcal{C}$ be an $[n, n-\rho, \rho+1]_q$ MDS code, for $q \geqslant n-1$, and let $\mathcal{H}(\mathcal{C})$ be its symmetric product code. The code $\mathcal{C}_{\mathcal{U}_2}$ is defined as follows,

$$\mathcal{C}_{\mathcal{U}_2} = \{G = (V_n, L_{\mathcal{U}}) \mid A_G \in \mathcal{H}(\mathcal{C})\}.$$

Notice that by Lemma 8 the dimension of code $\mathcal{H}(\mathcal{C})$ is $\binom{n-\rho+1}{2}$, and therefore by the definition of the code $\mathcal{C}_{\mathcal{U}_2}$, its dimension is $k_{\mathcal{U}} = \binom{n-\rho+1}{2}$. All other details for the correctness proof of Construction 3 are identical to the one of Theorem 7, and thus we only state here the next theorem.

Theorem 9. *For all ρ and n such that $\rho < n$, the code $\mathcal{C}_{\mathcal{U}_2}$ is an optimal undirected ρ-node-erasure-correcting code $\mathcal{U}$-$[n, k_{\mathcal{U}} = \binom{n-\rho+1}{2}]_{\mathbb{F}_q}$, where $q \geqslant n-1$.*

Similarly to the code $\mathcal{C}_{\mathcal{D}_1}$, the code $\mathcal{C}_{\mathcal{U}_2}$ is also systematic.

IV. Binary-Node-Erasure-Correcting Codes

In this section we study binary constructions of codes over directed and undirected graphs based upon the results by Roth from [14] and Schmidt from [16], [17].

An $[n \times n, k, \mu]$ linear array code $\mathcal{C}$ over a field $\mathbb{F}$ is a k-dimensional linear subspace of $n \times n$ matrices over $\mathbb{F}$, where the minimum rank of all nonzero matrices in $\mathcal{C}$ is at least μ. It was shown in [14] that such codes can correct $\mu - 1$ row or column erasures. Furthermore, the bound on such array codes states that $k \leqslant n(n - \mu + 1)$ [14]. In this section we present non-optimal binary constructions for undirected and directed codes over graphs based upon the results from [14], [16], [17].

A. Binary Construction of Codes Over Directed Graphs

A construction of binary $[n \times n, k, \mu]$ linear array codes where $k = n(n - \mu + 1)$ and $\mu = 2\rho + 1$ was shown in [14]. Based on these codes, we present the following construction of binary directed codes over graphs.

Construction 4 Let $\mathcal{C}$ be an $[n \times n, n(n-2\rho), 2\rho+1]$ binary optimal array code from [14], where $\rho < n/2$. The code over graphs $\mathcal{C}_{\mathcal{D}_3}$ is defined as follows,

$$\mathcal{C}_{\mathcal{D}_3} = \{G = (V_n, L_{\mathcal{D}}) \mid A_G \in \mathcal{C}\}.$$

Next the correctness of Construction 4 is proved.

Theorem 10. *For all $\rho < n/2$, the code $\mathcal{C}_{\mathcal{D}_3}$ is a linear binary directed ρ-node-erasure-correcting code $\mathcal{D}$-$[n, k_{\mathcal{D}} = n(n - 2\rho)]$.*

Proof: Notice that since the code $\mathcal{C}$ is linear, the code $\mathcal{C}_{\mathcal{D}_3}$ is also linear. Let G be a graph in the code $\mathcal{C}_{\mathcal{D}_3}$ and let A_G be its adjacency matrix. Assume some ρ nodes failed in G. The failure of these ρ nodes corresponds to erasure of the same ρ rows and columns in A_G. Since $A_G \in \mathcal{C}$ the minimum rank of A_G is $2\rho + 1$, and any 2ρ row or column erasures can be corrected. In particular, the erased ρ rows and ρ columns can be corrected as well, thereby correcting the ρ failed nodes. ■

Note that this construction does not provide optimal ρ-node-erasure-correcting codes since $r_{\mathcal{U}} = 2n\rho$, which does not meet the bound in (3). For example, for $\rho = 2$ the difference between the code redundancy and the bound is 4 redundancy bits.

The construction of binary optimal array codes $[n \times n, n(n - r), r + 1]$ from [14] has also a systematic construction, where the first $n - 2\rho$ rows of each matrix store the information bits and the last 2ρ rows store the redundancy bits. Therefore, we can use this family of codes also for the construction of systematic binary ρ-node-erasure-correcting codes over directed graphs $\mathcal{SD}$-$[n, k = n - 2\rho]$ for $\rho < n/2$. In this case, the number of redundancy edges will be $n^2 - (n - 2\rho)^2 = 4n\rho - 4\rho^2$. Therefore, the redundancy of this code is $(4n\rho - 4\rho^2) - (2n\rho - \rho^2) = 2n\rho - 3\rho^2$ far from optimality.

B. Binary Construction of Codes Over Undirected Graphs

A construction of binary $[n \times n, k, \mu]$ symmetric linear array codes where

$$k = \begin{cases} n(n-\mu+2)/2 & , n-\mu \text{ is even,} \\ (n+1)(n-\mu+1)/2 & , n-\mu \text{ is odd,} \end{cases}$$

was shown in [16]. Based on these codes, we present the following construction of binary undirected ρ-node-erasure-correcting codes.

Construction 5 Let $\mathcal{C}$ be an $[n \times n, k, \mu = 2\rho+1]$ symmetric binary array code from [16], where

$$k = \begin{cases} n(n-2\rho+1)/2 & , n \text{ is odd,} \\ (n+1)(n-2\rho)/2 & , n \text{ is even,} \end{cases}$$

and $\rho < n/2$. The code over graphs $\mathcal{C}_{\mathcal{U}_4}$ is defined as follows,

$$\mathcal{C}_{\mathcal{U}_4} = \{G = (V_n, L_{\mathcal{U}}) \mid A_G \in \mathcal{C}\}.$$

The proof of the correctness of Construction 5 is identical to the one of Theorem 10 and is stated in the next theorem.

Theorem 11. *For all $\rho < n/2$ and*

$$k_{\mathcal{U}} = \begin{cases} n(n-2\rho+1)/2 & , n \text{ is odd,} \\ (n+1)(n-2\rho)/2 & , n \text{ is even,} \end{cases}$$

the code $\mathcal{C}_{\mathcal{U}_4}$ is a linear binary undirected ρ-node-erasure-correcting code $\mathcal{U}$-$[n, k_{\mathcal{U}}]$.

This construction also does not provide optimal ρ-node-erasure-correcting codes since

$$r_{\mathcal{U}} = \begin{cases} n\rho & , n \text{ is odd,} \\ (n+1)\rho & , n \text{ is even,} \end{cases}$$

which does not achieve the bound in (4). For example, for $\rho = 2$ the difference between the code redundancy and the bound is one redundancy bit when n is odd and three bits when n is even.

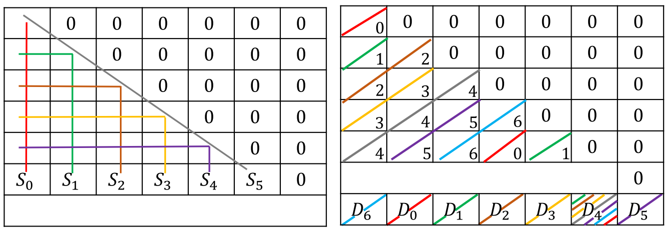
In Section III we saw constructions for optimal codes over graphs over a field of size at least $n - 1$, and in this section we saw binary constructions that do not provide optimal codes over graphs. Our next task is to achieve these two properties simultaneously, that is, optimal binary codes over graphs. In the next section we show how to accomplish this task for two node failures, when the number of nodes is a prime number. The general case for arbitrary number of node failures is left for future work.

V. Optimal Binary Undirected Double-Node-Erasure-Correcting Codes

In this section we present a construction of binary double-node-erasure-correcting codes for undirected graphs. We use the notation $\langle a \rangle_n$ to denote the value of $(a \bmod n)$.

Throughout this section we assume that $n \geqslant 5$ is a prime number. Let $G = (V_n, L_{\mathcal{U}})$ be a graph with n vertices. Let us define for $h \in [n-1]$

$$S_h = \begin{cases} \left\{ \langle v_h, v_\ell \rangle \mid \ell \in [n-1] \right\} & , h \in [n-2], \\ \left\{ \langle v_\ell, v_\ell \rangle \mid \ell \in [n-1] \right\} & , h = n-2. \end{cases} \quad (5)$$



(a) (b)

Fig. 3. The neighborhood and the diagonal sets. (a) Neighborhood parity paths. (b) Diagonal parity paths.

and for $m \in [n]$

$$D_m = \big\{\langle v_k, v_\ell\rangle \,|\, k,\ell \in [n]\setminus\{n-2\}, \langle k+\ell\rangle_n = m\big\} \cup \big\{\langle v_{n-1}, v_{n-2}\rangle\big\}. \tag{6}$$

Each set S_h where $h \in [n-2]$, will be used to represent the parity constraint on the neighborhood of node v_h, which correspond to row and column h in the adjacency matrix A_G. Similarly, for $m \in [n]$, each set D_m will represent parity constraints on the diagonals of A_G. We first show the following properties on the sets S_h and D_m, which their proof appear in Appendix A.

Claim 1. *For all $h \in [n-1]$, $|S_h| = n-1$ and for all $m \in [n]$, $|D_m| = \frac{n+1}{2}$.*

Example 3. The sets S_h, D_m for $n = 7$ are marked in Fig. 3. Note that the entries on lines with the same color belong to the same parity constraints.

Recall that for $t \in [n]$ the failure set F_t of the t-th node is its neighborhood set which we denote by $F_t = \{\langle v_t, v_\ell\rangle \mid \ell \in [n]\}$. The following connections between the sets S_h, D_m, F_t hold and will be used in the correctness of the construction we present in this section.

The following claim will be in use in the proof of Theorem 12 and its proof is given in Appendix B.

Claim 2. *The sets S_h, D_m, F_t satisfy the following properties.*

(a) For all distinct $h, i \in [n-2]$, $S_h \cap F_i = \{\langle v_h, v_i\rangle\}$.

(b) For all $i \in [n-2]$, $S_{n-2} \cap F_i = \{\langle v_i, v_i\rangle\}$.

(c) For all pairwise distinct $i, j, h \in [n-2]$, $S_h \cap (F_i \cup F_j) = \{\langle v_h, v_i\rangle, \langle v_h, v_j\rangle\}$.

(d) For all distinct $i, j \in [n-2]$, $S_{n-2} \cap (F_i \cup F_j) = \{\langle v_i, v_i\rangle, \langle v_j, v_j\rangle\}$.

(e) For all $i \in [n-2]$, $D_{\langle i-2\rangle_n} \cap F_i = \emptyset$ and for $i = n-1$, $D_{\langle n-3\rangle_n} \cap F_{n-1} = \{\langle v_{n-2}, v_{n-1}\rangle\}$.

(f) For all $i \in [n-2]$, $s \in [n] \setminus \{\langle i-2\rangle_n\}$, $D_s \cap F_i = \{\langle v_{\langle \boldsymbol{s-i}\rangle_n}, v_i\rangle\}$. For i such that $i = n-1$, $D_s \cap F_{n-1} = \{\langle v_{\langle s+1\rangle_n}, v_{n-1}\rangle, \langle v_{n-2}, v_{n-1}\rangle\}$

(g) For all distinct $i, j \in [n-2]$, $D_{\langle i+j\rangle_n} \cap F_j = \{\langle v_i, v_j\rangle\}$.

(h) For all distinct $i, j \in [n-2]$, $D_{\langle j-2\rangle_n} \cap (F_i \cup F_j) = \{\langle v_{\langle j-i-2\rangle_n}, v_i\rangle\}$

(i) For all distinct $i, j \in [n-2]$, $D_{\langle i+j\rangle_n} \cap (F_i \cup F_j) = \{\langle v_i, v_j\rangle\}$.

(j) For all $i \in [n-2]$, $D_{\langle i-2\rangle_n} \cap (F_i \cup F_{n-2}) = \{\langle v_{n-1}, v_{n-2}\rangle\}$.

We are now ready to present the construction of optimal binary undirected double-node-erasure-correcting codes.

Construction 6 For all $n \geqslant 5$ prime number, let $\mathcal{C}_{\mathcal{U}_5}$ be the following code.

$$\mathcal{C}_{\mathcal{U}_5} = \left\{ G = (V_n, L_{\mathcal{U}}) \left| \begin{array}{l} (a) \sum_{\langle v_i, v_j\rangle \in S_h} L_{\mathcal{U}}\langle v_i, v_j\rangle = 0, h \in [n-1] \\ (b) \sum_{\langle v_i, v_j\rangle \in D_m} L_{\mathcal{U}}\langle v_i, v_j\rangle = 0, m \in [n] \end{array} \right.\right\}.$$

Note that in this binary construction we have two sets of constraints, (a) and (b). The first set has $n-1$ constraints and we call each one of them constraint S_h, $h \in [n-1]$. Similarly, the second set has n constraints that will be called constraint D_m, $m \in [n]$. Note that the edge $\langle v_{n-1}, v_{n-2}\rangle$ appears in each of the diagonal sets, and therefore for $m \neq n-3$ the constraints D_m are dependent on the constraint D_{n-3}. We will need that in order to have successful decoding when the failed nodes are $i \in [n-2]$ and $j = n-2$, as will be shown for this case in the proof. Lastly, the correctness of this construction could be proved using Theorem 6 by showing that the minimum graph distance of this code is three, however, this will not provide a decoding algorithm as we present in the following proof.

Theorem 12. *The code $\mathcal{C}_{\mathcal{U}_5}$ is an optimal binary undirected double-node-erasure-correcting code.*

Proof: Assume that nodes $i, j \in [n]$, where $i < j$ are the failed nodes. We distinguish between the following three cases.

Case 1: $i \in [n-1]$, $j = n-1$. Using the S_h constraints, $h \in [n-1] \setminus \{i\}$, the edge set $F_i \setminus \{\langle v_i, v_{n-2}\rangle, \langle v_i, v_{n-1}\rangle\}$ can be corrected by

$$L_{\mathcal{U}}\langle v_i, v_h\rangle = \sum_{\langle v_k, v_h\rangle \in S_h \setminus \{\langle v_i, v_h\rangle\}} L_{\mathcal{U}}\langle v_k, v_h\rangle \quad : \quad h \neq n-2,$$

$$L_{\mathcal{U}}\langle v_i, v_i\rangle = \sum_{\langle v_k, v_k\rangle \in S_{n-2} \setminus \{\langle v_i, v_i\rangle\}} L_{\mathcal{U}}\langle v_k, v_k\rangle \quad : \quad h = n-2,$$

since for $h \neq i, n-2$ by Claim 2(a) $S_h \cap F_i = \{\langle v_h, v_i\rangle\}$, and for $h = n-2$ by Claim 2(b) $S_{n-2} \cap F_i = \{\langle v_i, v_i\rangle\}$. The constraint S_i then can be used to correct $\langle v_i, v_{n-2}\rangle$ by

$$L_{\mathcal{U}}\langle v_i, v_{n-2}\rangle = \sum_{\langle v_k, v_i\rangle \in S_i \setminus \{\langle v_i, v_{n-2}\rangle\}} L_{\mathcal{U}}\langle v_k, v_i\rangle.$$

Notice that F_{n-1} is the set of all the uncorrected edges left. By Claim 2(e), $D_{n-3} \cap F_{n-1} = \{\langle v_{n-2}, v_{n-1}\rangle\}$, so we first use the constraint D_{n-3} to correct the edge $\langle v_{n-2}, v_{n-1}\rangle$ by

$$L_{\mathcal{U}}\langle v_{n-2}, v_{n-1}\rangle = \sum_{\langle v_k, v_\ell\rangle \in D_{n-3} \setminus \{\langle v_{n-2}, v_{n-1}\rangle\}} L_{\mathcal{U}}\langle v_k, v_\ell\rangle.$$

By Claim 2(f) for $m \in [n] \setminus \{n-3\}$,

$$D_m \cap F_{n-1} = \{\langle v_{\langle m+1\rangle_n}, v_{n-1}\rangle, \langle v_{n-2}, v_{n-1}\rangle\},$$

and since the edge $\langle v_{n-2}, v_{n-1}\rangle$ is corrected, the D_m constraints can be used. Therefore, the remaining edges of the set F_{n-1} are corrected by

$$L_{\mathcal{U}}\langle v_{\langle m+1\rangle_n}, v_{n-1}\rangle = \sum_{\langle v_k, v_\ell\rangle \in D_m \setminus \{\langle v_{\langle m+1\rangle_n}, v_{n-1}\rangle\}} L_{\mathcal{U}}\langle v_k, v_\ell\rangle,$$

and that finishes the decoding of this case.

Case 2: $i \in [n-2]$, $j = n-2$. By Claim 2(j) $D_{\langle i-2\rangle_n} \cap (F_i \cup F_{n-2}) = \{\langle v_{n-1}, v_{n-2}\rangle\}$, so using the constraint $D_{\langle i-2\rangle_n}$, the edge $\langle v_{n-1}, v_{n-2}\rangle$ is corrected by

$$L_{\mathcal{U}}\langle v_{n-1}, v_{n-2}\rangle = \sum_{\langle v_k, v_\ell\rangle \in D_{\langle i-2\rangle_n} \setminus \{\langle v_{n-1}, v_{n-2}\rangle\}} L_{\mathcal{U}}\langle v_k, v_\ell\rangle.$$

By Claim 2(f) for all $m \in [n] \setminus \{\langle i-2\rangle_n\}$, $D_m \cap F_i = \{\langle v_{\langle m\text{-}i\rangle_n}, v_i\rangle\}$, and since $\langle v_{n-1}, v_{n-2}\rangle$ is the only edge that intersects between constraints D_m and F_{n-2}, $D_m \cap (F_i \cup F_{n-2}) = \{\langle v_{\langle m\text{-}i\rangle_n}, v_i\rangle, \langle v_{n-1}, v_{n-2}\rangle\}$. Since edge $\langle v_{n-1}, v_{n-2}\rangle$ is corrected, the edges in the set $F_i \setminus \{\langle v_i, v_{n-2}\rangle\}$ are corrected by the constraints D_m as follows,

$$L_{\mathcal{U}}\langle v_{\langle m\text{-}i\rangle_n}, v_i\rangle = \sum_{\langle v_k, v_\ell\rangle \in D_m \setminus \{\langle v_{\langle m\text{-}i\rangle_n}, v_i\rangle\}} L_{\mathcal{U}}\langle v_k, v_\ell\rangle.$$

Notice that $F_{n-2} \setminus \{\langle v_{n-2}, v_{n-1}\rangle\}$ is the set of all the uncorrected edges left. Thus, it is corrected using the S_h constraints, $h \in [n-1]$, similarly to the first case.

Case 3: $j < n-2$. In this case we show an explicit algorithm which decodes all the erased edges. First, we denote the *single parity syndromes* for $h \in [n-1] \setminus \{i, j\}$ by

$$\widehat{S}_h = \sum_{\langle v_k, v_\ell\rangle \in S_h \setminus (F_i \cup F_j)} L_{\mathcal{U}}\langle v_k, v_\ell\rangle,$$

and the *diagonal parity syndromes* for $m \in [n]$ by

$$\widehat{D}_m = \sum_{\langle v_k, v_\ell\rangle \in D_m \setminus (F_i \cup F_j)} L_{\mathcal{U}}\langle v_k, v_\ell\rangle.$$

Algorithm 1

1: $b_{prev} \leftarrow 0$
2: **for** $t = 0, 1, \ldots, x$ **do**
3: $s_1 \leftarrow \langle -d(t+1) - 2\rangle_n$
4: $s_2 \leftarrow \langle s_1 + j\rangle_n$
5: **if** $(s_1 \notin \{i, j, n-1\})$ **then**
6: $L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle \leftarrow \widehat{D}_{s_2} + b_{prev}$
7: $L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle \leftarrow \widehat{S}_{s_1} + L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle$
8: $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle$
9: **if** $(s_1 = j)$ **then**
10: $L_{\mathcal{U}}\langle v_j, v_j\rangle \leftarrow \widehat{D}_{s_2} + b_{prev}$
11: $L_{\mathcal{U}}\langle v_i, v_i\rangle \leftarrow \widehat{S}_{n-2} + L_{\mathcal{U}}\langle v_j, v_j\rangle$
12: $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_i, v_i\rangle$
13: **if** $s_1 = n-1$ **then**
14: $L_{\mathcal{U}}\langle v_{n-1}, v_j\rangle \leftarrow \widehat{D}_{s_2} + b_{prev}$
15: $b_{prev} \leftarrow 0$
16: **for** $t = 0, 1, \ldots, y$ **do**
17: $s_1 \leftarrow \langle d(t+1) - 2\rangle_n$
18: $s_2 \leftarrow \langle s_1 + i\rangle_n$
19: **if** $(s_1 \notin \{i, j, n-1\})$ **then**
20: $L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle \leftarrow \widehat{D}_{s_2} + b_{prev}$
21: $L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle \leftarrow \widehat{S}_{s_1} + L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle$
22: $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle$
23: **if** $(s_1 = i)$ **then**
24: $L_{\mathcal{U}}\langle v_i, v_i\rangle \leftarrow \widehat{D}_{s_2} + b_{prev}$
25: $L_{\mathcal{U}}\langle v_j, v_j\rangle \leftarrow \widehat{S}_{n-2} + L_{\mathcal{U}}\langle v_i, v_i\rangle$
26: $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_j, v_j\rangle$
27: **if** $s_1 = n-1$ **then**
28: $L_{\mathcal{U}}\langle v_{n-1}, v_i\rangle \leftarrow \widehat{D}_{s_2} + b_{prev}$

Let $d = \langle j\text{-}i\rangle_n$, $x = \langle -1 - d^{-1}\rangle_n$ and $y = \langle -1 + d^{-1}\rangle_n$. The decoding procedure is described in Algorithm 1.

In order to prove the correctness of the algorithm, we define an auxiliary parameter which describes the set of uncorrected edges at the beginning of each iteration of the algorithm. More specifically, denote $F^{(t)}$ as the set of uncorrected edges in the first loop and $\widetilde{F}^{(t)}$ in the second loop, so $F^{(0)} = F_i \cup F_j$ and $\widetilde{F}^{(0)} = F^{(x)}$. Denote by $s_1^{(t)}, s_2^{(t)}$ the values of s_1, s_2, respectively on iteration t in the first loop. The values of $\widetilde{s}_1^{(t)}$ and $\widetilde{s}_2^{(t)}$ will be defined similarly for the second loop.

The values of $s_1^{(t)}$ and $s_2^{(t)}$ are given by:

$$s_1^{(t)} = \langle -d(t+1) - 2\rangle_n, s_2^{(t)} = \langle s_1^{(t)} + j\rangle_n = \langle -dt + i - 2\rangle_n.$$

Similar expressions can be derived for $\widetilde{s}_1^{(t)}$ and $\widetilde{s}_2^{(t)}$. We also denote the following sets,

$$A = \{s_1^{(t_1)} | 0 \leqslant t_1 \leqslant x\}, B = \{\widetilde{s}_1^{(t_2)} | 0 \leqslant t_2 \leqslant y\}. \quad (7)$$

Before proving the correctness of the algorithm, we show some more useful properties while their proofs are deferred to Appendix C.

Claim 3. *The following properties hold:*

(a) $x \neq y$ *and* $x + y = n-2$.
(b) $s_1^{(x)} = \widetilde{s}_1^{(y)} = n-1$.
(c) $i, j \in A$ *or* $i, j \in B$ *but not in both.*
(d) $n-2 \notin A \cup B$.
(e) $A \cap B = \{n-1\}$.
(f) $|A| = x+1$ *and* $|B| = y+1$.
(g) $s_1^{(0)} \neq i$ *and* $\widetilde{s}_1^{(0)} \neq j$.
(h) For $0 < t \leqslant x$, $s_1^{(t-1)} = j$ *if and only if* $s_1^{(t)} = i$.
(i) For $0 < t \leqslant x$, $s_2^{(t)} \notin \{\langle i-2\rangle_n, \langle j-2\rangle_n\}$.

According to Claim 3(c), the variable s_1 in Algorithm 1 gets the values of i and j either in the first or in the second loop. Let us assume for the rest of the proof that this happens in the first loop, i.e., $i, j \in A$, while the second case is proved similarly. We are now ready to show the correctness of the first loop by induction while the proof for the second loop is very similar.

Lemma 13. *For all* $0 \leqslant t \leqslant x$, *the following properties hold:*

1) *If* $s_1^{(t)} \notin \{i, j, n-1\}$ *then* $D_{s_2^{(t)}} \cap F^{(t)} = \{\langle v_{s_1^{(t)}}, v_j\rangle\}$, $S_{s_1^{(t)}} \cap F^{(t)} = \{\langle v_{s_1^{(t)}}, v_i\rangle, \langle v_{s_1^{(t)}}, v_j\rangle\}$, *and the edges* $\langle v_{s_1^{(t)}}, v_j\rangle, \langle v_{s_1^{(t)}}, v_i\rangle$ *are corrected on the* t*-th iteration.*
2) *If* $s_1^{(t)} = j$ *then* $D_{s_2^{(t)}} \cap F^{(t)} = \{\langle v_j, v_j\rangle\}$, $S_{n-2} \cap F^{(t)} = \{\langle v_i, v_i\rangle, \langle v_j, v_j\rangle\}$ *and the edges* $\langle v_i, v_i\rangle, \langle v_j, v_j\rangle$ *are corrected on the* t*-th iteration.*
3) *If* $s_1^{(t)} = n-1$ *then* $D_{s_2^{(t)}} \cap F^{(t)} = \{\langle v_{n-1}, v_j\rangle\}$ *and the edge* $\langle v_{n-1}, v_j\rangle$ *is corrected on the* t*-th iteration.*

Proof: We prove this claim by induction on t. First note that for all $t \in [n]$, it holds that $s_1^{(t)} \neq n-2$ as it was proved in Claim 3(d).

Base: For $t = 0$ we have $s_1^{(0)} = \langle -d-2\rangle_n$, $s_2^{(0)} = \langle i-2\rangle_n$, and $F^{(0)} = F_i \cup F_j$. By Claim 3(g), $s_1^{(0)} \neq i$. We first prove the case where $s_1^{(0)} \notin \{j, n-1\}$. Hence, we need to show that,

1) $D_{\langle i-2\rangle_n} \cap (F_i \cup F_j) = \{\langle v_{\langle -d-2\rangle_n}, v_j\rangle\}$,
2) $S_{\langle -d-2\rangle_n} \cap (F_i \cup F_j) = \{\langle v_{\langle -d-2\rangle_n}, v_i\rangle, \langle v_{\langle -d-2\rangle_n}, v_j\rangle\}$.
3) The edges $\langle v_{\langle -d-2\rangle_n}, v_j\rangle$ and $\langle v_{\langle -d-2\rangle_n}, v_i\rangle$ are corrected on this iteration.

The proof consists of the following observations:

- According to Claim 2(h) we deduce that

  $$D_{\langle i-2\rangle_n} \cap (F_i \cup F_j) = \{\langle v_{\langle i-j-2\rangle_n}, v_j\rangle\} = \{\langle v_{\langle -d-2\rangle_n}, v_j\rangle\}$$

  and therefore the edge $\langle v_{\langle -d-2\rangle_n}, v_j\rangle$ is corrected in Step 6 according to the constraint $D_{\langle i-2\rangle_n}$, therefore, $L_{\mathcal{U}}\langle v_{\langle -d-2\rangle_n}, v_j\rangle = \widehat{D}_{\langle i-2\rangle_n}$.
- According to Claim 2(c) we get

  $$S_{\langle -d-2\rangle_n} \cap (F_i \cup F_j) = \{\langle v_{\langle -d-2\rangle_n}, v_i\rangle, \langle v_{\langle -d-2\rangle_n}, v_j\rangle\},$$

and therefore the edge $\langle v_{\langle -d-2\rangle_n}, v_i\rangle$ is corrected in Step 7 according to the constraint $S_{\langle -d-2\rangle_n}$, by

$$L_{\mathcal{U}}\langle v_{\langle -d-2\rangle_n}, v_i\rangle = \widehat{S}_{\langle -d-2\rangle_n} + L_{\mathcal{U}}\langle v_{\langle -d-2\rangle_n}, v_j\rangle.$$

Notice that Steps 6,10 and 14 are identical. Therefore, if $s_1^{(0)}$ equals to $j, n-1$, the edge $\langle v_j, v_j\rangle$, $\langle v_{n-1}, v_j\rangle$ is corrected in Step 10,14 according to the constraint $D_{\langle i-2\rangle_n}$, respectively. If $s_1^{(0)} = j$, according to Claim 2(d) we get

$$S_{n-2} \cap (F_i \cup F_j) = \{\langle v_i, v_i\rangle, \langle v_j, v_j\rangle\},$$

and therefore the edge $\langle v_i, v_i\rangle$ is corrected in Step 11 according to the constraint S_{n-2}, by

$$L_{\mathcal{U}}\langle v_i, v_i\rangle = \widehat{S}_{n-2} + L_{\mathcal{U}}\langle v_j, v_j\rangle.$$

Step: Assume that the induction assumption holds for $t-1$, where $t \leqslant x$ and we prove its correctness for t. In this case, by Claim 3(b) and Claim 3(f), only $s_1^{(x)}$ in A is equal to $n-1$, so we have that $s_1^{(t-1)} \neq n-1$. If $s_1^{(t-1)} \notin \{i, j\}$, we assume that the edges $\langle v_{s_1^{(t-1)}}, v_j\rangle$ and $\langle v_{s_1^{(t-1)}}, v_i\rangle$, were corrected on the $t-1$ iteration. If $s_1^{(t-1)} = j$, by Claim 3(h) it holds if and only if $s_1^{(t)} = i$. Notice that the algorithm do nothing on this iteration and it can be assumed that the edges $\langle v_j, v_j\rangle$ and $\langle v_i, v_i\rangle$ were corrected on the $t-1$ iteration. Therefore, we left to analyze the case where $s_1^{(t-1)} \notin \{j, n-1\}$. We consider the following cases:

1) $s_1^{(t)} \notin \{j, n-1\}$: By Claim 3(i) $s_2^{(t)} \notin \{\langle i-2\rangle_n, \langle j-2\rangle_n\}$ and by Claim 2(f), we deduce that

$$\begin{aligned} D_{s_2^{(t)}} \cap (F_i \cup F_j) &= \{\langle v_{\langle s_2^{(t)}-i\rangle_n}, v_i\rangle, \langle v_{\langle s_2^{(t)}-j\rangle_n}, v_j\rangle\} \\ &= \{\langle v_{s_1^{(t-1)}}, v_i\rangle, \langle v_{s_1^{(t)}}, v_j\rangle\}. \end{aligned}$$

By the induction assumption $\langle v_{s_1^{(t-1)}}, v_i\rangle$ was corrected, so,

$$D_{s_2^{(t)}} \cap (F_i^{(t)} \cup F_j^{(t)}) = \{\langle v_{s_1^{(t)}}, v_j\rangle\},$$

and the edge $\langle v_{s_1^{(t)}}, v_j\rangle$ is successfully corrected in Step 6 by $D_{s_2^{(t)}}$ constraint. Furthermore, since $s_1^{(t)} \neq n-1$, by Claim 2(c),

$$S_{s_1^{(t)}} \cap (F_i \cup F_j) = \{\langle v_{s_1^{(t)}}, v_i\rangle, \langle v_{s_1^{(t)}}, v_j\rangle\},$$

so it holds that $S_{s_1^{(t)}} \cap (F_i^{(t)} \cup F_j^{(t)}) = \{\langle v_{s_1^{(t)}}, v_i\rangle, \langle v_{s_1^{(t)}}, v_j\rangle\}$ and therefore the edge $\langle v_{s_1^{(t)}}, v_i\rangle$ can be successfully corrected in Step 7 by constraint $S_{s_1^{(t)}}$ and the value of $L_{\mathcal{U}}\langle v_{s_1^{(t)}}, v_j\rangle$. Notice that in this case $s_1^{(t-1)}$ can also be equal to i.

2) $s_1^{(t)} = j$ or $s_1^{(t)} = n-1$: Since Steps 6,10 and 14 are identical, we first correct the edge $\langle v_{s_1^{(t)}}, v_j\rangle$ by the $D_{s_2^{(t)}}$ constraint. In case that $s_1^{(t)} = j$, by Claim 2(d), $S_{n-2} \cap (F_i \cup F_j) = \{\langle v_i, v_i\rangle, \langle v_j, v_j\rangle\}$ so it holds that $S_{n-2} \cap (F_i^{(t)} \cup F_j^{(t)}) = \{\langle v_i, v_i\rangle, \langle v_j, v_j\rangle\}$. Therefore the edge $\langle v_i, v_i\rangle$ is corrected in Step 11 by the S_{n-2} constraint and the value of $L_{\mathcal{U}}\langle v_j, v_j\rangle$. ■

A similar lemma for the second loop is stated as follows. We omit its proof since it is very similar to the one of Lemma 13.

Lemma 14. *For all $0 \leqslant t \leqslant y$, by assumption that $i, j \notin B$ the following properties hold:*

1) If $\widetilde{s}_1^{(t)} \neq n-1$ then $D_{\widetilde{s}_2^{(t)}} \cap F^{(t)} = \{\langle v_{\widetilde{s}_1^{(t)}}, v_j\rangle\}$, $S_{\widetilde{s}_1^{(t)}} \cap F^{(t)} = \{\langle v_{\widetilde{s}_1^{(t)}}, v_i\rangle, \langle v_{\widetilde{s}_1^{(t)}}, v_j\rangle\}$, and the edges $\langle v_{\widetilde{s}_1^{(t)}}, v_j\rangle$, $\langle v_{\widetilde{s}_1^{(t)}}, v_i\rangle$ are corrected on the t-th iteration.

2) If $\widetilde{s}_1^{(t)} = n-1$ then $D_{\widetilde{s}_2^{(t)}} \cap F^{(t)} = \{\langle v_{n-1}, v_j\rangle\}$ and the edge $\langle v_{n-1}, v_j\rangle$ is corrected on the t-th iteration.

Let V_1, V_2 be the set of edges which were corrected in the first, second loop, respectively. Hence,

$$\begin{aligned} V_1 &= \{\langle v_{s_1}, v_i\rangle, \langle v_{s_1}, v_j\rangle : s_1 \in A \setminus \{n-1\}\} \cup \{\langle v_{n-1}, v_j\rangle\}, \\ V_2 &= \{\langle v_{\widetilde{s}_1}, v_j\rangle, \langle v_{\widetilde{s}_1}, v_i\rangle : \widetilde{s}_1 \in B \setminus \{n-1\}\} \cup \{\langle v_{n-1}, v_i\rangle\}. \end{aligned}$$

We also define $V = V_1 \cup V_2$, and prove the following claim in Appendix D.

Claim 4. *The following properties hold:*

(a) $V_1 \cap V_2 = \emptyset$.
(b) $|V| = 2n-4$.
(c) $\langle v_i, v_j\rangle \notin V$.
(d) $\langle v_{n-2}, v_i\rangle \notin V$ and $\langle v_{n-2}, v_j\rangle \notin V$.

Lastly, at the end of the algorithm the following property on the set of uncorrected edges holds.

Claim 5. *At the end of the algorithm,*

$$\widetilde{F}^{(y)} = \{\langle v_i, v_j\rangle, \langle v_{n-2}, v_i\rangle, \langle v_{n-2}, v_j\rangle\}$$

Proof: Since the number of erased edges is $2n-1$, we get that

$$|\widetilde{F}^{(y)}| = (2n-1) - |V| = (2n-1) - (2n-4) = 3.$$

By Claim 4(c) and Claim 4(d), the edges that were not decoded yet are $\{\langle v_i, v_j\rangle, \langle v_{n-2}, v_i\rangle, \langle v_{n-2}, v_j\rangle\}$ and therefore,

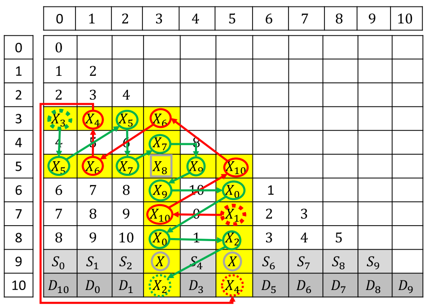
$$\widetilde{F}^{(y)} = \{\langle v_i, v_j\rangle, \langle v_{n-2}, v_i\rangle, \langle v_{n-2}, v_j\rangle\}.$$

■

According to Claim 2(i) $D_{\langle i+j\rangle_n} \cap (F_i \cup F_j) = \{\langle v_i, v_j\rangle\}$ and therefore the edge $\langle v_i, v_j\rangle$ can be reconstructed by the constraint $D_{\langle i+j\rangle_n}$, that is, $L_{\mathcal{U}}\langle v_i, v_j\rangle = \widehat{D}_{\langle i+j\rangle_n}$. Since the only uncorrected edges of nodes v_i and v_j are $\langle v_i, v_{n-2}\rangle$, $\langle v_j, v_{n-2}\rangle$, they are corrected by the constraints S_i and S_j. The number of constraints of this code is $2n-1$, which meets the bound in (4) so it is an optimal code. As mentioned before, each optimal code is also a systematic code, and thus this code is systematic. ■

Note that for the systematic version of the code $\mathcal{C}_{\mathcal{U}_5}$, where the last two nodes are the redundancy nodes, we sometimes refer the first node among them by the *single parity node* and the second one by the *diagonal parity node*. The decoding algorithm presented in the proof of Theorem 12 is demonstrated in the next example.

Example 4. In this example we show a decoding scheme of $\mathcal{C}_{\mathcal{U}_5}$ code $\mathcal{SU}$-[11, 9]. We consider the case where the failed nodes are v_3 and v_5, that is, $i = 3$, $j = 5$. Therefore $d = 2$ and $x = 4$, $y = 5$. The undirected graph is represented by a lower-triangle-adjacency matrix. The yellow cells of the matrix



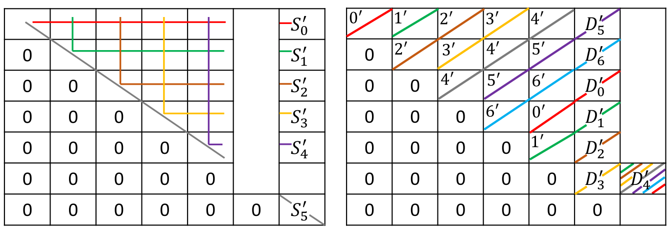
(a)

First Loop (Red)			
s_2	s_1	$\langle v_{s_1}, v_j\rangle$	$\langle v_{s_1}, v_i\rangle$
1	7	$\langle v_7, v_5\rangle$	$\langle v_7, v_3\rangle$
10	5	$\langle v_5, v_5\rangle$	$\langle v_3, v_3\rangle$
8	3	–	–
6	1	$\langle v_1, v_5\rangle$	$\langle v_1, v_3\rangle$
4	10	$\langle v_{10}, v_5\rangle$	

Second Loop (Green)			
s_2	s_1	$\langle v_{s_1}, v_i\rangle$	$\langle v_{s_1}, v_j\rangle$
3	0	$\langle v_0, v_3\rangle$	$\langle v_0, v_5\rangle$
5	2	$\langle v_2, v_3\rangle$	$\langle v_2, v_5\rangle$
7	4	$\langle v_4, v_3\rangle$	$\langle v_4, v_5\rangle$
9	6	$\langle v_6, v_3\rangle$	$\langle v_6, v_5\rangle$
0	8	$\langle v_8, v_3\rangle$	$\langle v_8, v_5\rangle$
2	10	$\langle v_{10}, v_3\rangle$	

(b)

Fig. 4. A decoding scheme of an optimal binary undirected double-node-erasure-correcting code $\mathcal{SU}$-[11, 9]. (a) Simulation of the algorithm. (b) Corrected edge order.



(a) (b)

Fig. 5. The neighborhood and the diagonal sets. (a) Neighborhood parity paths. (b) Diagonal parity paths.

represent the erased edges. As mentioned before, the decoding procedure in Algorithm 1 corrects all the erased edges using two loops. The first loop is represented by Steps 1-14 and the second loop is represented by Steps 15-28. In each iteration of both loops we calculate s_1 and s_2 that we show in the tables of Figure 4. Thus, we get the sets A and B, that is, $A = \{7, 5, 3, 1, 10\}$ and $B = \{0, 2, 4, 6, 8, 10\}$. Therefore, the first loop starts with the edge $\langle v_7, v_5\rangle$, since $s_1^{(0)} = \langle -d-2\rangle_n = \langle -2-2\rangle_{11} = 7$ and ends with the edge $\langle v_{10}, v_5\rangle$, since $s_1^{(4)} = \langle -d(x+1)-2\rangle_n = \langle -2\cdot 5-2\rangle_{11} = 10$. Similarly, the second loop starts with the edge $\langle v_3, v_0\rangle$ and ends with the edge $\langle v_{10}, v_3\rangle$. Notice that $5, 3 \in A$, so the first loop corrects the self loop edges $\langle v_5, v_5\rangle$ and $\langle v_3, v_3\rangle$. At the end of this algorithm the edge $\langle v_5, v_3\rangle, \langle v_9, v_3\rangle, \langle v_9, v_5\rangle$ that is marked with gray, is corrected with the constraint D_8, S_3, S_5, respectively.

We finish this section by showing another construction of systematic binary undirected double-node-erasure-correcting code $\mathcal{SU}$-$[n, n-2]$ which is very similar to the code from Construction 6. Here, we present this construction by showing the constraints of graphs on their upper-triangle-adjacency matrices. We have this construction since it will be used in Section VI for the construction of optimal binary directed double-node-erasure-correcting codes $\mathcal{SD}$-$[n, n-2]$. This construction is almost a symmetric reflection of Construction 6 with respect to the main diagonal. However, we had to introduce only a single modification in which we changed between the roles of the redundancy nodes v_{n-2} and v_{n-1}.

Let $G = (V_n, L_{\mathcal{U}})$ be a graph with n vertices. Let us define for $h \in [n-1]$

$$S'_h = \begin{cases} \{\langle v_h, v_\ell\rangle \mid \ell \in [n] \setminus \{n-2\}\} & , h \in [n-2], \\ \{\langle v_\ell, v_\ell\rangle \mid \ell \in [n] \setminus \{n-2\}\} & , h = n-2, \end{cases} \quad (8)$$

and for $m \in [n]$,

$$D'_m = \{\langle v_k, v_\ell\rangle | k, \ell \in [n-1]\}, \langle k+\ell\rangle_n = m\} \cup \{\langle v_{n-2}, v_{n-1}\rangle\}. \quad (9)$$

As before, the sets S'_h for $h \in [n-1]$ and D'_m for $m \in [n]$, will be used to represent parity constraints on the upper-triangle-adjacency matrix of each graph.

Example 5. The sets S'_h, D'_m for $n = 7$ are marked in Fig. 5. Entries on lines with the same color belong to the same parity constraint.

Our second construction of optimal binary undirected double-node-erasure-correcting codes $\mathcal{SU}$-$[n, n-2]$ works as follows.

Construction 7 For all $n \geqslant 5$ prime number let $\mathcal{C}_{\mathcal{U}_6}$ be the following code over graphs,

$$\mathcal{C}_{\mathcal{U}_6} = \left\{ G = (V_n, L_{\mathcal{U}}) \middle| \begin{array}{l} (a) \sum_{\langle v_i, v_j\rangle \in S'_h} L_{\mathcal{U}}\langle v_i, v_j\rangle = 0, h \in [n-1] \\ (b) \sum_{\langle v_i, v_j\rangle \in D'_m} L_{\mathcal{U}}\langle v_i, v_j\rangle = 0, m \in [n] \end{array} \right\}.$$

Theorem 15. *The code $\mathcal{C}_{\mathcal{U}_6}$ is an optimal binary undirected double-node-erasure-correcting code.*

We will not prove here the correctness of the code $\mathcal{C}_{\mathcal{U}_6}$ since its construction is very similar to one of the code $\mathcal{C}_{\mathcal{U}_5}$. However, note that when constructing the code $\mathcal{C}_{\mathcal{U}_6}$, we switched the roles of the last two redundancy nodes such that the first node is the diagonal parity node and the second node is the single parity node. We still present here a decoding algorithm of this code for the more challenging case when the failed nodes are v_i, v_j and $i, j \in [n-2]$. Its correctness is similar to the one of Algorithm 1 and is thus omitted.

First, we denote the *single parity syndromes* for $h \in [n-1] \setminus \{i, j\}$ by

$$\widehat{S}'_h = \sum_{\langle v_k, v_\ell\rangle \in S'_h \setminus (F_i \cup F_j)} L_{\mathcal{U}}\langle v_k, v_\ell\rangle,$$

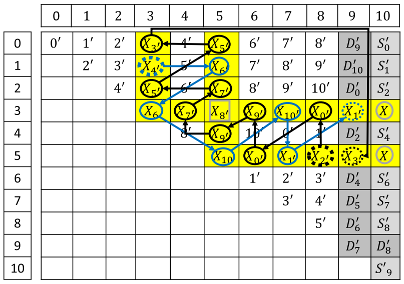
and the *diagonal parity syndromes* for $m \in [n]$ by

$$\widehat{D}'_m = \sum_{\langle v_k, v_\ell\rangle \in D'_m \setminus (F_i \cup F_j)} L_{\mathcal{U}}\langle v_k, v_\ell\rangle.$$

Let $x' = \langle -1 + d^{-1}\rangle_n$ and $y' = \langle -1 - d^{-1}\rangle_n$. The decoding procedure for this case is described in Algorithm 2.

Algorithm 2

$b_{prev} \leftarrow 0$
for $t = 0, 1, \ldots, x'$ **do**
  $s_1 \leftarrow \langle -d(t+1) - 1\rangle_n$
  $s_2 \leftarrow \langle s_1 + j\rangle_n$
  if ($s_1 \notin \{i, j, n-2\}$) **then**
    $L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle \leftarrow \widehat{D}'_{s_2} + b_{prev}$
    $L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle \leftarrow \widehat{S}'_{s_1} + L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle$
    $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle$
  if ($s_1 = j$) **then**
    $L_{\mathcal{U}}\langle v_j, v_j\rangle \leftarrow \widehat{D}'_{s_2} + b_{prev}$
    $L_{\mathcal{U}}\langle v_i, v_i\rangle \leftarrow \widehat{S}'_{n-2} + L_{\mathcal{U}}\langle v_j, v_j\rangle$
    $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_i, v_i\rangle$
  if $s_1 = n-2$ **then**
    $L_{\mathcal{U}}\langle v_{n-2}, v_j\rangle \leftarrow \widehat{D}'_{s_2} + b_{prev}$
$b_{prev} \leftarrow 0$
for $t = 0, 1, \ldots, y'$ **do**
  $s_1 \leftarrow \langle d(t+1) - 1\rangle_n$
  $s_2 \leftarrow \langle s_1 + i\rangle_n$
  if ($s_1 \notin \{i, j, n-2\}$) **then**
    $L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle \leftarrow \widehat{D}'_{s_2} + b_{prev}$
    $L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle \leftarrow \widehat{S}'_{s_1} + L_{\mathcal{U}}\langle v_{s_1}, v_i\rangle$
    $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_{s_1}, v_j\rangle$
  if ($s_1 = i$) **then**
    $L_{\mathcal{U}}\langle v_i, v_i\rangle \leftarrow \widehat{D}'_{s_2} + b_{prev}$
    $L_{\mathcal{U}}\langle v_j, v_j\rangle \leftarrow \widehat{S}'_{n-2} + L_{\mathcal{U}}\langle v_i, v_i\rangle$
    $b_{prev} \leftarrow L_{\mathcal{U}}\langle v_j, v_j\rangle$
  if $s_1 = n-2$ **then**
    $L_{\mathcal{U}}\langle v_{n-2}, v_i\rangle \leftarrow \widehat{D}'_{s_2} + b_{prev}$



(a)

First Loop (Black)			
s_2	s_1	$\langle v_{s_1}, v_j\rangle$	$\langle v_{s_1}, v_i\rangle$
2	8	$\langle v_8, v_5\rangle$	$\langle v_8, v_3\rangle$
0	6	$\langle v_6, v_5\rangle$	$\langle v_6, v_3\rangle$
9	4	$\langle v_4, v_5\rangle$	$\langle v_4, v_3\rangle$
7	2	$\langle v_2, v_5\rangle$	$\langle v_2, v_3\rangle$
5	0	$\langle v_0, v_5\rangle$	$\langle v_0, v_3\rangle$
3	9	$\langle v_9, v_5\rangle$	

Second Loop (Blue)			
s_2	s_1	$\langle v_{s_1}, v_i\rangle$	$\langle v_{s_1}, v_j\rangle$
4	1	$\langle v_1, v_3\rangle$	$\langle v_1, v_5\rangle$
6	3	$\langle v_3, v_3\rangle$	$\langle v_5, v_5\rangle$
8	5	–	–
10	7	$\langle v_7, v_3\rangle$	$\langle v_7, v_5\rangle$
1	9	$\langle v_9, v_3\rangle$	

(b)

Fig. 6. A decoding scheme of an optimal binary undirected double-node-erasure-correcting code $\mathcal{SU}$-[11, 9]. (a) Simulation of the algorithm. (b) Corrected edge order.

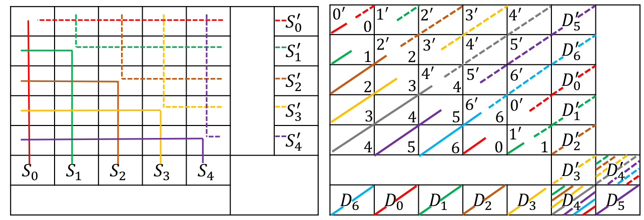
The decoding algorithm presented in the proof of Theorem 15 is demonstrated in the next example.

Example 6. In this example we show a decoding scheme of the code $\mathcal{C}_{\mathcal{U}_6}$ for $n = 11$. We consider the case where the failed nodes are v_3 and v_5, that is, $i = 3, j = 5$. Therefore $d = 2$ and $x = 5, y = 4$. The undirected graph is represented by an upper-triangle-adjacency matrix. The yellow cells of the matrix represent the erased edges. As mentioned before, the decoding procedure in Algorithm 2 corrects all erased edges using two loops. The first loop is represented by Steps 1-14 and the second loop is represented by Steps 15-28. In each iteration of both loops we calculate s_1 and s_2 that we show in the tables of Figure 6. Thus, we get the sets A and B to be $A = \{8, 6, 4, 2, 0, 9\}$ and $B = \{1, 3, 5, 7, 9\}$. Therefore, the first loop starts with the edge $\langle v_8, v_5\rangle$, since $s_1^{(0)} = \langle -d-1\rangle_n = \langle -2-1\rangle_{11} = 8$ and ends with the edge $\langle v_9, v_5\rangle$, since $s_1^{(5)} = \langle -d(x+1)-1\rangle_n = \langle -2\cdot 6-1\rangle_{11} = 9$. Similarly, the second loop starts with the edge $\langle v_1, v_3\rangle$ and ends with the edge $\langle v_9, v_3\rangle$. Notice that $3, 5 \in B$, so the second loop corrects the self loop edges $\langle v_3, v_3\rangle$ and $\langle v_5, v_5\rangle$. At the end of this algorithm the edge $\langle v_5, v_3\rangle, \langle v_{10}, v_3\rangle, \langle v_{10}, v_5\rangle$ that is marked with gray, is corrected using the constraint D'_8, S'_3, S'_5, respectively.

Note, that the code $\mathcal{C}_{\mathcal{U}_6}$ is also an optimal and therefore it is systematic.

VI. Optimal Binary Directed Double-Node-Erasure-Correcting Codes

In this section we combine between Constructions 6 and 7 in order to generate an optimal binary directed double-node-erasure-correcting code. The main idea here is to use Construction 6 in order to correct the backward edges (v_i, v_j) for $i > j$, i.e. the edges in the lower part of the matrix, and Construction 7 for the correction of the forward edges (v_i, v_j) for $i < j$ which are the edges in the upper part of the matrix. However, since the self loops are involved in both of these parts, we will have to carefully interleave between the two constructions. In particular, this dependency affects also the decoding of the two constructions which will have to be combined as well. Throughout this section we assume that $n \geqslant 5$ is a prime number.



(a) (b)

Fig. 7. The neighborhood and the diagonal sets. (a) Neighborhood Parity Paths. (b) Diagonal parity paths.

Let $G = (V_n, L_{\mathcal{D}})$ be a directed graph with n vertices and let $G_2 = (V_n, L_{\mathcal{U}})$ be an undirected graph. We use the same definitions of the sets S_h, S'_h, D_m, D'_m, $h \in [n-2], m \in [n]$ from (5), (8), (6), (9), respectively, and let F_t, $t \in [n]$ be a failure set.

For $i, j \in [n]$, not necessarily distinct, let $\langle v_i, v_j\rangle^{\downarrow}$ be the edge directed from $v_{\max\{i,j\}}$ to $v_{\min\{i,j\}}$, i.e., $\langle v_i, v_j\rangle^{\downarrow} = (v_{\max\{i,j\}}, v_{\min\{i,j\}})$, and similarly $\langle v_i, v_j\rangle^{\uparrow} = (v_{\min\{i,j\}}, v_{\max\{i,j\}})$ is the edge directed from $v_{\min\{i,j\}}$ to $v_{\max\{i,j\}}$. Next we define the constraint sets for G.

For $h \in [n-2]$ the *neighborhood-edge sets* $S_h^{\downarrow}, S_h^{\uparrow}$ are defined by

$$S_h^{\downarrow} = \{\langle v_i, v_j\rangle^{\downarrow} | \langle v_i, v_j\rangle \in S_h\},$$
$$S_h^{\uparrow} = \{\langle v_i, v_j\rangle^{\uparrow} | \langle v_i, v_j\rangle \in S'_h\}.$$

Furthermore, for $m \in [n]$ the *diagonal-edge sets* $D_m^{\downarrow}, D_m^{\uparrow}$ are defined by

$$D_m^{\downarrow} = \{\langle v_i, v_j\rangle^{\downarrow} | \langle v_i, v_j\rangle \in D_m\},$$
$$D_m^{\uparrow} = \{\langle v_i, v_j\rangle^{\uparrow} | \langle v_i, v_j\rangle \in D'_m\},$$

and lastly for $t \in [n]$ the *failure-edge sets* $F_t^{\downarrow}, F_t^{\uparrow}$ are defined by

$$F_t^{\downarrow} = \{\langle v_i, v_j\rangle^{\downarrow} | \langle v_i, v_j\rangle \in F_t\},$$
$$F_t^{\uparrow} = \{\langle v_i, v_j\rangle^{\uparrow} | \langle v_i, v_j\rangle \in F_t\}.$$

Example 7. The sets $S_h^{\downarrow}, S_h^{\uparrow}, D_m^{\downarrow}, D_m^{\uparrow}$ for $n = 7$ are marked in Fig. 7. Entries on lines with the same color belong to the same parity constraint.

The following claim for the directed case is very similar to the corresponding one from Claim 2. Thus, we omit its proof.

Claim 6.

(a) For all distinct $h, i \in [n-2]$, $S_h^{\uparrow} \cap F_i^{\uparrow} = \{\langle v_h, v_i\rangle^{\uparrow}\}$.

(b) For all $i \in [n-2]$, $D^{\downarrow}_{\langle i-2\rangle_n} \cap (F_i^{\downarrow} \cup F_{n-2}^{\downarrow}) = \{(v_{n-1}, v_{n-2})\}$.

(c) For all $i \in [n-2]$, $s \in [n] \setminus \{\langle i-2\rangle_n\}$, $D_s^{\downarrow} \cap F_i^{\downarrow} = \{\langle v_{\langle s-i\rangle_n}, v_i\rangle^{\downarrow}\}$.

(d) For all distinct $i, j \in [n-2]$, $D^{\downarrow}_{\langle i+j\rangle_n} \cap F_j^{\downarrow} = \{(v_j, v_i)\}$ and $D^{\uparrow}_{\langle i+j\rangle_n} \cap F_j^{\uparrow} = \{(v_i, v_j)\}$.

We are now ready to present the construction of optimal binary directed double-node-erasure-correcting codes.

Construction 8 For all $n \geqslant 5$ prime number let $\mathcal{C}_{\mathcal{D}_7}$ be the following code.

$$\mathcal{C}_{\mathcal{D}_7} = \left\{ G=(V_n, L) \left| \begin{array}{l} (a)\sum_{(v_i,v_j)\in S_h^{\downarrow}} L_{\mathcal{D}}(v_i,v_j)=0, h\in[n-2] \\ (b)\sum_{(v_i,v_j)\in D_m^{\downarrow}} L_{\mathcal{D}}(v_i,v_j)=0, m\in[n] \\ (c)\sum_{(v_i,v_j)\in S_h^{\uparrow}} L_{\mathcal{D}}(v_i,v_j)=0, h\in[n-2] \\ (d)\sum_{(v_i,v_j)\in D_m^{\uparrow}} L_{\mathcal{D}}(v_i,v_j)=0, m\in[n] \end{array} \right. \right\}.$$

In this binary construction we did not use the constraints that were derived from the two sets S_{n-2} and S'_{n-2} (i.e., the constraints on the main diagonal). We will show that an encoding scheme of this code is also systematic. Assume the first $n-2$ nodes carry the information symbols on their edges. It can be verified that for all $i \in [n-2]$ the encoding with each of the constraints $S_i^{\downarrow}, S_i^{\uparrow}, D_{n-3}^{\downarrow}$ and $D_{n-3}^{\uparrow}$ is possible since each of them encodes its appropriate redundancy edge using a simple parity code. For all $m \in [n] \setminus \{n-3\}$ the encoding with constraints $D_m^{\downarrow}$ and $D_m^{\uparrow}$, is also possible since the edges (v_{n-1}, v_{n-2}) and (v_{n-2}, v_{n-1}) were encoded with respect to $D_{n-3}^{\downarrow}$ and $D_{n-3}^{\uparrow}$. Hence, in the next proof for the correctness of the construction we will refer to it as a systematic construction.

Theorem 16. *The code $\mathcal{C}_{\mathcal{D}_7}$ is an optimal binary directed double-node-erasure-correcting code.*

Proof: Assume that the nodes $i, j \in [n], i < j$ are failed. We will show the correctness of this construction by splitting it into three cases.

Case 1: $i = n-2, j = n-1$. In this case the information edges have not been erased, therefore, we can use the encoding rules of $\mathcal{C}_{\mathcal{D}_7}$ to correct the failed redundancy nodes.

Case 2: $i \in [n-2]$, $j = n-2$. In this case, by Claim 6(a) for $h \in [n-2] \setminus \{i\}$, $S_h^{\uparrow} \cap F_i^{\uparrow} = \{\langle v_h, v_i\rangle^{\uparrow}\}$ so the following $n-3$ edges connected to the i-th node can be corrected by constraint (c) of Construction 8, that is,

$$L_{\mathcal{D}}\langle v_h, v_i\rangle^{\uparrow} = \sum_{(v_k,v_\ell)\in S_h^{\uparrow}\setminus\{\langle v_h,v_i\rangle^{\uparrow}\}} L_{\mathcal{D}}(v_k, v_\ell).$$

Notice that we corrected $n-3$ information edges on the upper-triangle-adjacency matrix of the graph, where the edges (v_i, v_i), (v_i, v_{n-2}) and (v_i, v_{n-1}) are not corrected yet. From Claim 6(b) we get that $D_{\langle i-2\rangle_n}^{\downarrow} \cap (F_i^{\downarrow} \cup F_{n-2}^{\downarrow}) = \{(v_{n-1}, v_{n-2})\}$. Therefore the edge (v_{n-1}, v_{n-2}) can be corrected using constraint (b) of Construction 8,

$$L_{\mathcal{D}}(v_{n-1}, v_{n-2}) = \sum_{(v_k,v_\ell)\in D_{\langle i-2\rangle_n}^{\downarrow}\setminus\{(v_{n-1},v_{n-2})\}} L_{\mathcal{D}}(v_k, v_\ell).$$

By Claim 6(c) for all $m \in [n] \setminus \{\langle i-2\rangle_n\}$, $D_m^{\downarrow} \cap F_i^{\downarrow} = \{\langle v_{\langle m\text{-}i\rangle_n}, v_i\rangle^{\downarrow}\}$. Since (v_{n-1}, v_{n-2}) is the only edge that intersects between constraints $D_m^{\downarrow}$ and $F_{n-2}^{\downarrow}$, we get that $D_m^{\downarrow} \cap (F_i^{\downarrow} \cup F_{n-2}^{\downarrow}) = \{\langle v_{\langle m\text{-}i\rangle_n}, v_i\rangle^{\downarrow}, (v_{n-1}, v_{n-2})\}$. Therefore, the following edges connected to the node v_i are decoded by constraint (b) of Construction 8,

$$L_{\mathcal{D}}\langle v_{\langle m\text{-}i\rangle_n}, v_i\rangle^{\downarrow} = \sum_{(v_k,v_\ell)\in D_m^{\downarrow}\setminus\{\langle v_{\langle m\text{-}i\rangle_n},v_i\rangle^{\downarrow}\}} L_{\mathcal{D}}(v_k, v_\ell).$$

It can be verified that we corrected another n edges on the lower-triangle-adjacency matrix of the graph, where there are no uncorrected information edges left. Thus, the remaining edges can be successfully decoded according to the encoding rules of the code. For $j = n-1$ the proof is very similar and thus we omit its details.

Case 3: The *neighborhood syndromes* $\widehat{S}_h^{\downarrow}, \widehat{S}_h^{\uparrow}$ are defined by

$$\widehat{S}_h^{\downarrow} = \sum_{(v_k,v_\ell)\in S_h^{\downarrow}\setminus(F_i^{\downarrow}\cup\ F_j^{\downarrow})} L_{\mathcal{D}}(v_k, v_\ell),$$
$$\widehat{S}_h^{\uparrow} = \sum_{(v_k,v_\ell)\in S_h^{\uparrow}\setminus(F_i^{\uparrow}\cup\ F_j^{\uparrow})} L_{\mathcal{D}}(v_k, v_\ell),$$

and the *diagonal syndromes* $\widehat{D}_m^{\downarrow}, \widehat{D}_m^{\uparrow}$ are defined by

$$\widehat{D}_m^{\downarrow} = \sum_{(v_k,v_\ell)\in D_m^{\downarrow}\setminus(F_i^{\downarrow}\cup F_j^{\downarrow})} L_{\mathcal{D}}(v_k, v_\ell),$$
$$\widehat{D}_m^{\uparrow} = \sum_{(v_k,v_\ell)\in D_m^{\uparrow}\setminus(F_i^{\uparrow}\cup F_j^{\uparrow})} L_{\mathcal{D}}(v_k, v_\ell).$$

Let $d = \langle \text{j-i}\rangle_n$, $x = \langle -1-d^{-1}\rangle_n$, $y = \langle -1+d^{-1}\rangle_n$, $x' = \langle -1+d^{-1}\rangle_n$ and $y' = \langle -1-d^{-1}\rangle_n$. The decoding procedure for the code $\mathcal{C}_{\mathcal{D}_7}$ in this case is described in Algorithm 3.

This algorithm consists of four loops marked as Loop I, II, III, and IV. For $Y \in \{\text{I, II, III, IV}\}$, denote by $s_{1,Y}^{(t)}$ the value of the variable s_1 on iteration t of Loop Y. These values of $s_{1,Y}^{(t)}$ are given by:

$$s_{1,\text{I}}^{(t)} = \langle -d(t+1)-2\rangle_n, s_{1,\text{II}}^{(t)} = \langle d(t+1)-2\rangle_n,$$
$$s_{1,\text{III}}^{(t)} = \langle -d(t+1)-1\rangle_n, s_{1,\text{IV}}^{(t)} = \langle d(t+1)-1\rangle_n.$$

Next, we denote the following four sets:

$$A = \{s_{1,\text{I}}^{(t)} : t \in [x+1]\}, B = \{s_{1,\text{II}}^{(t)} : t \in [y+1]\},$$
$$A' = \{s_{1,\text{III}}^{(t')} : t' \in [x'+1]\}, B' = \{s_{1,\text{IV}}^{(t')} : t \in [y'+1]\}.$$

Claim 7. *The indices i, j satisfy the following property: $i, j \in A \cap B'$ or $i, j \in A' \cap B$, but not in both.*

Proof: Notice that sets A and B are defined similarly to (7). In Claim 3(c), it was stated that $i, j \in A$ or $i, j \in B$, but not in both, and it is possible to show that the same property holds for A' and B'. Without loss of generality, let us assume that $i, j \in A$. Let $0 \leqslant t < x-1$ be a step in which $s_{1,\text{I}}^{(t)} = j$ and therefore $s_{1,\text{I}}^{(t+1)} = i$. Therefore we can calculate $s_{1,\text{IV}}^{(x-(t+1))}$ by,

$$\begin{aligned} & s_{1,\text{IV}}^{(x-(t+1))} - j = s_{1,\text{IV}}^{(x-(t+1))} - s_{1,\text{I}}^{(t)} = \\ & \langle d(x-(t+1)+1)-1\rangle_n - \langle -d(t+1)-2\rangle_n = \\ & \langle d(x-t)+d(t+1)+1\rangle_n = \langle d(x+1)+1\rangle_n = \\ & \langle d((-1-d^{-1})+1)+1\rangle_n = -1+1 = 0, \end{aligned}$$

Algorithm 3

Loop I

1: $b_{prev} \leftarrow 0$
2: **for** $t = 0, 1, \ldots, x$ **do**
3: $s_1 \leftarrow \langle -d(t+1) - 2 \rangle_n$
4: $s_2 \leftarrow \langle s_1 + j \rangle_n$
5: **if** $(s_1 \notin \{i, j, n-1\})$ **then**
6: $L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\downarrow} \leftarrow \widehat{D}^{\downarrow}_{s_2} + b_{prev}$
7: $L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\downarrow} \leftarrow \widehat{S}^{\downarrow}_{s_1} + L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\downarrow}$
8: $b_{prev} \leftarrow L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\downarrow}$
9: **if** $(s_1 = j)$ **then**
10: $L_{\mathcal{D}}(v_j, v_j) \leftarrow \widehat{D}^{\downarrow}_{s_2} + b_{prev}$
11: Wait until (v_i, v_i) is corrected.
12: $b_{prev} \leftarrow L_{\mathcal{D}}(v_i, v_i)$
13: **if** $s_1 = n-1$ **then**
14: $L_{\mathcal{D}}(v_{n-1}, v_j) \leftarrow \widehat{D}^{\downarrow}_{s_2} + b_{prev}$

Loop II

15: $b_{prev} \leftarrow 0$
16: **for** $t = 0, 1, \ldots, y$ **do**
17: $s_1 \leftarrow \langle d(t+1) - 2 \rangle_n$
18: $s_2 \leftarrow \langle s_1 + i \rangle_n$
19: **if** $(s_1 \notin \{i, j, n-1\})$ **then**
20: $L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\downarrow} \leftarrow \widehat{D}^{\downarrow}_{s_2} + b_{prev}$
21: $L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\downarrow} \leftarrow \widehat{S}^{\downarrow}_{s_1} + L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\downarrow}$
22: $b_{prev} \leftarrow L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\downarrow}$
23: **if** $(s_1 = i)$ **then**
24: $L_{\mathcal{D}}(v_i, v_i) \leftarrow \widehat{D}^{\downarrow}_{s_2} + b_{prev}$
25: Wait until (v_j, v_j) is corrected.
26: $b_{prev} \leftarrow L_{\mathcal{D}}(v_j, v_j)$
27: **if** $s_1 = n-1$ **then**
28: $L_{\mathcal{D}}(v_{n-1}, v_i) \leftarrow \widehat{D}^{\downarrow}_{s_2} + b_{prev}$

Loop III

29: $b_{prev} \leftarrow 0$
30: **for** $t = 0, 1, \ldots, x'$ **do**
31: $s_1 \leftarrow \langle -d(t+1) - 1 \rangle_n$
32: $s_2 \leftarrow \langle s_1 + j \rangle_n$
33: **if** $(s_1 \notin \{i, j, n-2\})$ **then**
34: $L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\uparrow} \leftarrow \widehat{D}^{\uparrow}_{s_2} + b_{prev}$
35: $L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\uparrow} \leftarrow \widehat{S}^{\uparrow}_{s_1} + L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\uparrow}$
36: $b_{prev} \leftarrow L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\uparrow}$
37: **if** $(s_1 = j)$ **then**
38: $L_{\mathcal{D}}(v_j, v_j) \leftarrow \widehat{D}^{\uparrow}_{s_2} + b_{prev}$
39: Wait until (v_i, v_i) is corrected.
40: $b_{prev} \leftarrow L_{\mathcal{D}}(v_i, v_i)$
41: **if** $s_1 = n-2$ **then**
42: $L_{\mathcal{D}}(v_j, v_{n-2}) \leftarrow \widehat{D}^{\uparrow}_{s_2} + b_{prev}$

Loop IV

43: $b_{prev} \leftarrow 0$
44: **for** $t = 0, 1, \ldots, y'$ **do**
45: $s_1 \leftarrow \langle d(t+1) - 1 \rangle_n$
46: $s_2 \leftarrow \langle s_1 + i \rangle_n$
47: **if** $(s_1 \notin \{i, j, n-2\})$ **then**
48: $L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\uparrow} \leftarrow \widehat{D}^{\uparrow}_{s_2} + b_{prev}$
49: $L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\uparrow} \leftarrow \widehat{S}^{\uparrow}_{s_1} + L_{\mathcal{D}}\langle v_{s_1}, v_i \rangle^{\uparrow}$
50: $b_{prev} \leftarrow L_{\mathcal{D}}\langle v_{s_1}, v_j \rangle^{\uparrow}$
51: **if** $(s_1 = i)$ **then**
52: $L_{\mathcal{D}}(v_i, v_i) \leftarrow \widehat{D}^{\uparrow}_{s_2} + b_{prev}$
53: Wait until (v_j, v_j) is corrected.
54: $b_{prev} \leftarrow L_{\mathcal{D}}(v_j, v_j)$
55: **if** $s_1 = n-2$ **then**
56: $L_{\mathcal{D}}(v_i, v_{n-2}) \leftarrow \widehat{D}^{\uparrow}_{s_2} + b_{prev}$

and therefore $s_{1,\text{IV}}^{(x-(t+1))} = j$. By definition of $s_{1,\text{IV}}^{(t)}$ we can see that

$$s_{1,\text{IV}}^{(x-(t+2))} = s_{1,\text{IV}}^{(x-(t+1))} - d = i,$$

where $t+2 \leqslant x$ and therefore $i, j \in B'$. The opposite direction is proved similarly. ■

The decoding Algorithm 3 for this case combines Algorithm 1 and Algorithm 2, where Algorithm 1 is used to decode the lower-triangle-adjacency matrix and Algorithm 2 is used to decode the upper-triangle-adjacency matrix. However, since we did not use the constraints of the two sets S_{n-2} and S'_{n-2} on the main diagonal, we had to replace Step 11, 25 in Algorithm 1, Algorithm 2 with the command *wait until* (v_i, v_i) *is corrected*, *wait until* (v_j, v_j) *is corrected*, respectively. According to Claim 7, the indices i, j satisfy $i, j \in A \cap B'$ or $i, j \in A' \cap B$ but not both. Without loss of generality, assume that $i, j \in A \cap B'$. Therefore, in this case, Loops II and III of Algorithm 3 will not be affected by the main diagonal constraint. This holds since the edges (v_i, v_i) and (v_j, v_j) are not corrected in these two loops as the conditions in Steps 23 and 37 will not hold. Hence, these two loops operate and succeed exactly as done in Algorithm 1 and Algorithm 2. This does not hold for Loops I and IV. Namely, Loop I, IV operates exactly as Algorithm 1, Algorithm 2 until Loop I, IV reaches Step 11, 53, respectively. Here we notice that according to Algorithm 1, in Step 11, the algorithm was supposed to correct the edge (v_i, v_i) according to the constraint on the mail diagonal. Similarly, in Step 53, the algorithm was supposed to correct the edge (v_j, v_j) according to the constraint on the mail diagonal. However, since the edge (v_i, v_i) is corrected in Loop IV and the edge (v_j, v_j) is corrected in Loop I, all we need to do in Step 11 is to wait for the edge (v_i, v_i) to be corrected and in the same way in Step 53 for the edge (v_j, v_j) to be corrected. Then, the rest of these two loops proceed to correct the remaining edges as done in Algorithm 1 and Algorithm 2.

Lastly, from Claim 6(d), $D^{\downarrow}_{\langle i+j \rangle_n} \cap F^{\downarrow}_j = \{(v_j, v_i)\}$ and $D^{\uparrow}_{\langle i+j \rangle_n} \cap F^{\uparrow}_j = \{(v_i, v_j)\}$, so the last two information edges (v_j, v_i) and (v_i, v_j) are corrected by constraints $\widehat{D}^{\downarrow}_{\langle i+j \rangle_n}$ and $\widehat{D}^{\uparrow}_{\langle i+j \rangle_n}$, respectively. Since all of the information edges were corrected, we can correct the remaining uncorrected redundancy edges (v_{n-2}, v_i),(v_{n-2}, v_j),(v_i, v_{n-1}) and (v_j, v_{n-1}) using our encoding rules. ■

Notice, that the number of constraints of this code is $4n - 4$, which meets the bound in (3) so it is an optimal code.

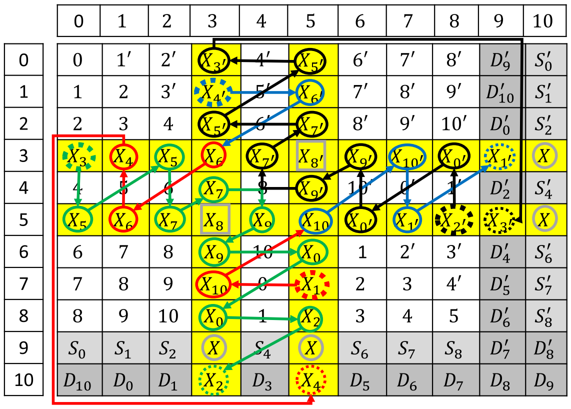
The decoding algorithm presented in the proof of Theorem 16 is demonstrated in the next example.

Example 8. In this example we show a decoding scheme of the code $\mathcal{C}_{\mathcal{D}_7}$ for $n = 11$. We consider the case where the failed nodes are v_3 and v_5, that is, $i = 3, j = 5$. Therefore $d = 2$ and $x = y' = 4, x' = y = 5$. The decoding procedure in Algorithm 3 corrects all the erased edges using four loops, where Loop I, II, III, IV is represented by Steps 1-14, 15-28, 29-42 Steps 43-56, respectively. We use here the lower-triangle-adjacency matrix for Loop I (red) and Loop II (green) and the upper-triangle-adjacency matrix for Loop III (black) and Loop IV (blue). The yellow cells of the matrix represent the erased edges. In each iteration of both loops we calculate the values of $s_{1,\text{I}}, s_{2,\text{I}}, s_{1,\text{II}}, s_{2,\text{II}}, s_{1,\text{III}}, s_{2,\text{III}}, s_{1,\text{IV}}$ and $s_{2,\text{IV}}$, that we show in the tables of Figure 8. Thus, we get the sets A, B, A' and B', where $A = \{7, 5, 3, 1, 10\}$, $B = \{0, 2, 4, 6, 8, 10\}$, $A' = \{8, 6, 4, 2, 0, 9\}$ and $B' = \{1, 3, 5, 7, 9\}$. Loop I starts with the edge (v_7, v_5), and ends with the edge (v_{10}, v_5) and Loop II starts with the edge (v_3, v_0) and ends with the edges (v_{10}, v_3). Similarly, Loop III starts with the edge (v_5, v_8), and ends with the edge (v_5, v_9) and finally Loop IV starts with the edge (v_1, v_3) and ends with the edge (v_3, v_9). Loop I, IV corrects the self loop (v_5, v_5), (v_3, v_3), respectively. At the end of this algorithm, the edge (v_5, v_3), (v_9, v_3), (v_9, v_5) that is marked with gray in the lower-triangle-adjacency matrix, is corrected using the constraint $D^{\downarrow}_8, S^{\downarrow}_3, S^{\downarrow}_5$, respectively. Similarly, the edge (v_3, v_5), (v_3, v_{10}), (v_5, v_{10}) that is marked with gray in the upper-triangle-adjacency matrix, is corrected using the constraint $D^{\uparrow}_8, S^{\uparrow}_3, S^{\uparrow}_5$, respectively.

VII. Bounds on Optimal Undirected ρ-Node-Erasure-Correcting Codes Over $\mathbb{F}_q$

In this section we study necessary conditions on the existence of optimal undirected ρ-node-erasure-correcting codes over $\mathbb{F}_q$ with n nodes. For the special case of $\rho = n - 2$ we will show a necessary and sufficient condition and explicitly find the number of such codes.

Every linear code over undirected graphs $\mathcal{U}$-$[n, k_{\mathcal{U}}]_{\mathbb{F}_q}$ can be represented by a generator matrix $\mathbf{G}$ of dimensions



(a)

First Loop (Red)			
s_2	s_1	$\langle v_{s_1}, v_j\rangle^{\downarrow}$	$\langle v_{s_1}, v_i\rangle^{\downarrow}$
1	7	$\langle v_7, v_5\rangle^{\downarrow}$	$\langle v_7, v_3\rangle^{\downarrow}$
10	5	$\langle v_5, v_5\rangle^{\downarrow}$	–
8	3	–	–
6	1	$\langle v_1, v_5\rangle^{\downarrow}$	$\langle v_1, v_3\rangle^{\downarrow}$
4	10	$\langle v_{10}, v_5\rangle^{\downarrow}$	

Second Loop (Green)			
s_2	s_1	$\langle v_{s_1}, v_i\rangle^{\downarrow}$	$\langle v_{s_1}, v_j\rangle^{\downarrow}$
3	0	$\langle v_0, v_3\rangle^{\downarrow}$	$\langle v_0, v_5\rangle^{\downarrow}$
5	2	$\langle v_2, v_3\rangle^{\downarrow}$	$\langle v_2, v_5\rangle^{\downarrow}$
7	4	$\langle v_4, v_3\rangle^{\downarrow}$	$\langle v_4, v_5\rangle^{\downarrow}$
9	6	$\langle v_6, v_3\rangle^{\downarrow}$	$\langle v_6, v_5\rangle^{\downarrow}$
0	8	$\langle v_8, v_3\rangle^{\downarrow}$	$\langle v_8, v_5\rangle^{\downarrow}$
2	10	$\langle v_{10}, v_3\rangle^{\downarrow}$	

(b)

Third Loop (Black)			
s_2	s_1	$\langle v_{s_1}, v_j\rangle^{\uparrow}$	$\langle v_{s_1}, v_i\rangle^{\uparrow}$
2	8	$\langle v_8, v_5\rangle^{\uparrow}$	$\langle v_8, v_3\rangle^{\uparrow}$
0	6	$\langle v_6, v_5\rangle^{\uparrow}$	$\langle v_6, v_3\rangle^{\uparrow}$
9	4	$\langle v_4, v_5\rangle^{\uparrow}$	$\langle v_4, v_3\rangle^{\uparrow}$
7	2	$\langle v_2, v_5\rangle^{\uparrow}$	$\langle v_2, v_3\rangle^{\uparrow}$
5	0	$\langle v_0, v_5\rangle^{\uparrow}$	$\langle v_0, v_3\rangle^{\uparrow}$
3	9	$\langle v_9, v_5\rangle^{\uparrow}$	

Forth Loop (Blue)			
s_2	s_1	$\langle v_{s_1}, v_i\rangle^{\uparrow}$	$\langle v_{s_1}, v_j\rangle^{\uparrow}$
4	1	$\langle v_1, v_3\rangle^{\uparrow}$	$\langle v_1, v_5\rangle^{\uparrow}$
6	3	$\langle v_3, v_3\rangle^{\uparrow}$	–
8	5	–	–
10	7	$\langle v_7, v_3\rangle^{\uparrow}$	$\langle v_7, v_5\rangle^{\uparrow}$
1	9	$\langle v_9, v_3\rangle^{\uparrow}$	

(c)

Fig. 8. A decoding scheme of an optimal binary directed double-node-erasure-correcting code $\mathcal{SD}$-[11, 9]. (a) Simulation of the algorithm. (b) Lower tringle corrected edge order. (c) Upper tringle corrected edge order.

$k_{\mathcal{U}} \times \binom{n+1}{2}$ over $\mathbb{F}_q$. We denote the columns of the generator matrix $\mathbf{G}$ by the indices of the set $\{(i,j) \in [n]^2 \mid i \geqslant j\}$, in their lexicographic order, so the column indexed by (i,j) is $\boldsymbol{g}_{i,j}$. For all $1 \leqslant \ell \leqslant n$, let S_ℓ be the set of all subsets of $[n]$ of size ℓ, that is,

$$S_\ell = \{B \mid B \subseteq [n], |B| = \ell\}. \tag{10}$$

For each $B \in S_\ell$, denote the column set V_B by,

$$V_B = \{\boldsymbol{g}_{i,j} | (i,j) \in B^2, i \geqslant j\}. \tag{11}$$

Clearly, the size of V_B, where $B \in S_\ell$, is $|V_B| = \binom{\ell+1}{2}$. The following lemma states a necessary and sufficient condition on the generator matrix of codes over undirected graphs to be optimal codes.

Lemma 17. *Let $\mathcal{C}_{\mathcal{U}}$ be a linear code over undirected graphs $\mathcal{U}$-$[n, k_{\mathcal{U}} = \binom{k+1}{2}]_{\mathbb{F}_q}$ and let $\mathbf{G}$ be its generator matrix. Then, $\mathcal{C}_{\mathcal{U}}$ is an optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$ if and only if for all $B \in S_k$, the columns of V_B are linearly independent.*

Proof: Let $\boldsymbol{u} = (u_1, u_2, \ldots, u_{\binom{k+1}{2}}) \in \mathbb{F}_q^{\binom{k+1}{2}}$ be an information vector encoded with the matrix $\mathbf{G}$. Assume that there was an erasure of $\rho = n - k$ nodes and let $B \in S_k$ be the set of k remaining nodes. Let $c_1, c_2, \ldots, c_{\binom{k+1}{2}}$ be the information symbols on the edges of the k remaining nodes in their lexicographic order. Similarly, let $\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_{\binom{k+1}{2}}$ be the columns of V_B in their lexicographic order. Finding the information vector $\boldsymbol{u} = (u_1, u_2, \ldots, u_{\binom{k+1}{2}})$ is achieved by solving the following equations system

$$[u_1, u_2, \ldots, u_{\binom{k+1}{2}}] \cdot [\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_{\binom{k+1}{2}}] = [c_1, c_2, \ldots, c_{\binom{k+1}{2}}],$$

that has a unique solution if and only if the columns of V_B are linearly independent. ∎

A. The $\rho = n - 2$ Case

In this section we show necessary and sufficient conditions for the existence of optimal undirected $(n-2)$-node-erasure-correcting codes over $\mathbb{F}_q$, and we also find the number of such codes.

Theorem 18. *For all positive integer $n \geqslant 3$ and prime power q, there exists an optimal undirected $(n-2)$-node-erasure-correcting code over $\mathbb{F}_q$ if and only if*

$$q^2 + q + 2 > n, \tag{12}$$

and in this case, the number of such codes is

$$q^{2\binom{n}{2}} (q-1)^{\binom{n+1}{2}} \frac{(q^2+q+1)!}{(q^2+q+1-n)!}.$$

Proof: Let $\mathcal{C}$ be an optimal code over undirected graphs $\mathcal{U}$-$[n,3]_{\mathbb{F}_q}$, and let $\mathbf{G}$ be its generator matrix. In this case, for $0 \leqslant i < j \leqslant n-1$ each column (i,j) of $\mathbf{G}$ is a vector from $\mathbb{F}_q^3$. According to Lemma 17, for all $i, j \in [n]$, the columns $\boldsymbol{g}_{i,i}, \boldsymbol{g}_{i,j}, \boldsymbol{g}_{j,j}$ are linearly independent. We will first find the number of possible columns for $\boldsymbol{g}_{i,i}$ and then for $\boldsymbol{g}_{i,j}$. For all distinct $i, j \in [n]$, the columns $\boldsymbol{g}_{i,i}, \boldsymbol{g}_{j,j}$ have to be linearly independent. First, for $\boldsymbol{g}_{0,0}$ we have $q^3 - 1$ options, all the vectors of $\mathbb{F}_q^3$ except for the zero vector. Then, for $1 \leqslant i \leqslant n-1$ we deduce that the number of valid options for $\boldsymbol{g}_{i,i}$ is

$$(q^3 - 1) - (q-1)i,$$

since it cannot be a linear combination of any previous column $\boldsymbol{g}_{k,k}$ for $0 \leqslant k < i$. Hence, for $i = n-1$ we require that $(q^3-1) - (q-1)(n-1) > 0$, that is,

$$q^2 + q + 2 > n,$$

which is a necessary condition for the existence of the code $\mathcal{C}$.

Next, for all $0 \leqslant i < j \leqslant n-1$ the column $\boldsymbol{g}_{i,j}$ cannot be linearly dependent on the columns $\boldsymbol{g}_{i,i}$ and $\boldsymbol{g}_{j,j}$, and therefore there are

$$(q^3 - 1) - (q^2 - 1) = q^2(q-1)$$

options to choose the column $\boldsymbol{g}_{i,j}$. Together, we conclude that the number of such codes will be the composition of all possible options, that is,

$$\begin{aligned}&[q^2(q-1)]^{\binom{n}{2}}\prod_{i=0}^{n-1}[(q^3-1)-(q-1)i]\\=&[q^2(q-1)]^{\binom{n}{2}}\prod_{i=0}^{n-1}[(q-1)(q^2+q+1)-(q-1)i]\\=&[q^2(q-1)]^{\binom{n}{2}}\prod_{i=0}^{n-1}[(q-1)(q^2+q+1-i)]\\=&[q^2(q-1)]^{\binom{n}{2}}(q-1)^n\prod_{i=0}^{n-1}[(q^2+q+1-i)]\\=&q^{2\binom{n}{2}}(q-1)^{\binom{n+1}{2}}\frac{(q^2+q+1)!}{(q^2+q+1-n)!}.\end{aligned}$$

Since for $n=q^2+q+1$ the number of such codes is a positive number, the condition in (12) is necessary and sufficient. ■

B. Arbitrary ρ

In this section we study a sufficient condition on the existence of optimal undirected ρ-node-erasure-correcting codes over $\mathbb{F}_q$, where $\rho = n-k$. For the rest of this section, we assume that k is even, $t=k/2$, and we let $\mathcal{C}_{\mathcal{U}}$ be an optimal undirected $(n-k)$-node-erasure-correcting code $\mathcal{U}$-$[n,\binom{k+1}{2}]_{\mathbb{F}_q}$, and $\mathbf{G}$ is its generator matrix. In order to find a necessary condition for the existence of $\mathcal{C}_{\mathcal{U}}$, we find a lower bound on the number of vectors in the set $\bigcup_{B\in S_t}\operatorname{span}V_B$, which is then translated into an upper bound on the value of n, since $\bigcup_{B\in S_t}\operatorname{span}V_B\subseteq\mathbb{F}_q^{\binom{k+1}{2}}$. We first prove the following claim.

Claim 8. *For all two distinct $B_1,B_2\subseteq[n]$, such that $|B_1\cup B_2|\leqslant k$, the columns of the set $V_{B_1}\cup V_{B_2}$ are linearly independent and*

$$\left(\operatorname{span}V_{B_1}\cap\operatorname{span}V_{B_2}\right)=\operatorname{span}\left(V_{B_1}\cap V_{B_2}\right).$$

Proof: According to Lemma 17, for all $B\in S_k$, the columns of V_B are linearly independent, and this property holds for any set B of size at most k. For all $B_1,B_2\subseteq[n]$, such that $|B_1\cup B_2|\leqslant k$, the columns of $V_{B_1\cup B_2}$ are linearly independent and in particular also the columns of the set $V_{B_1}\cup V_{B_2}$.

To prove the second part of this claim, first note that $V_{B_1}\cap V_{B_2}\subseteq V_{B_1}$ and hence $\operatorname{span}\left(V_{B_1}\cap V_{B_2}\right)\subseteq\operatorname{span}V_{B_1}$ and similarly $\operatorname{span}\left(V_{B_1}\cap V_{B_2}\right)\subseteq\operatorname{span}V_{B_2}$, that is,

$$\operatorname{span}\left(V_{B_1}\cap V_{B_2}\right)\subseteq\left(\operatorname{span}V_{B_1}\cap\operatorname{span}V_{B_2}\right).$$

Next, assume that $\boldsymbol{v}\in\left(\operatorname{span}V_{B_1}\cap\operatorname{span}V_{B_2}\right)$. Denote the vector set of V_{B_1},V_{B_2} by $\{\boldsymbol{u}_0,\boldsymbol{u}_1,\dots,\boldsymbol{u}_{|V_{B_1}|-1}\}$, $\{\boldsymbol{w}_0,\boldsymbol{w}_1,\dots,\boldsymbol{w}_{|V_{B_2}|-1}\}$, respectively. Therefore, there are coefficients $\alpha_0,\alpha_1,\dots,\alpha_{|V_{B_1}|-1}$ over $\mathbb{F}_q$ not all of them zero, and coefficients $\beta_0,\beta_1,\dots,\beta_{|V_{B_2}|-1}$ over $\mathbb{F}_q$ not all of them zero, such that,

$$\boldsymbol{v}=\sum_{i=0}^{|V_{B_1}|-1}\alpha_i\boldsymbol{u}_i=\sum_{i=0}^{|V_{B_2}|-1}\beta_i\boldsymbol{w}_i,$$

or equivalently,

$$\sum_{i=0}^{|V_{B_1}|-1}\alpha_i\boldsymbol{u}_i-\sum_{i=0}^{|V_{B_2}|-1}\beta_i\boldsymbol{w}_i=0.$$

Since $|B_1\cup B_2|\leqslant k$, the columns of the set $V_{B_1}\cup V_{B_2}$ are linearly independent. Since $\sum_{i=0}^{|V_{B_1}|-1}\alpha_i\boldsymbol{u}_i-\sum_{i=0}^{|V_{B_2}|-1}\beta_i\boldsymbol{w}_i=0$, we deduce that for all $\boldsymbol{u}_i\in V_{B_1}\setminus V_{B_2}$, $\alpha_i=0$ and for all $\boldsymbol{w}_i\in V_{B_2}\setminus V_{B_1}$, $\beta_i=0$. Therefore, $\boldsymbol{v}\in\operatorname{span}\left(V_{B_1}\cap V_{B_2}\right)$. ■

Next we will prove the following Lemma.

Lemma 19. *For all r such that $2\leqslant r\leqslant|S_t|$, and r distinct sets $B_1,B_2,\dots,B_r\in S_t$,*

$$\left(\bigcap_{1\leqslant i\leqslant r}\operatorname{span}V_{B_i}\right)=\operatorname{span}\left(\bigcap_{1\leqslant i\leqslant r}V_{B_i}\right).$$

Proof: We prove this lemma by induction on the value of r. The base case was already proved in Claim 8.

The step case for $r>2$ is proved as follows. Suppose that

$$\left(\bigcap_{1\leqslant i\leqslant r-1}\operatorname{span}V_{B_i}\right)=\operatorname{span}\left(\bigcap_{1\leqslant i\leqslant r-1}V_{B_i}\right),$$

and we will prove that

$$\left(\bigcap_{1\leqslant i\leqslant r}\operatorname{span}V_{B_i}\right)=\operatorname{span}\left(\bigcap_{1\leqslant i\leqslant r}V_{B_i}\right).$$

Since

$$\left(\bigcap_{1\leqslant i\leqslant r}\operatorname{span}V_{B_i}\right)=\left(\bigcap_{1\leqslant i\leqslant r-1}\operatorname{span}V_{B_i}\right)\cap\operatorname{span}V_{B_r}$$

we use the induction assumption to get that

$$\left(\bigcap_{1\leqslant i\leqslant r-1}\operatorname{span}V_{B_i}\right)\cap\operatorname{span}V_{B_r}=\operatorname{span}\left(\bigcap_{1\leqslant i\leqslant r-1}V_{B_i}\right)\cap\operatorname{span}V_{B_r},$$

and now we apply Claim 8 on the sets $\bigcap_{1\leqslant i\leqslant r-1}V_{B_i}$ and V_{B_r} to get that

$$\operatorname{span}\left(\bigcap_{1\leqslant i\leqslant r-1}V_{B_i}\right)\cap\operatorname{span}V_{B_r}=\operatorname{span}\left(\bigcap_{1\leqslant i\leqslant r}V_{B_i}\right),$$

which concludes the proof. ■

Next, for all $1\leqslant r\leqslant|S_t|$, let $f(n,t,r,s)$ be the number of options to choose r sets from S_t where their intersection is of size s, that it,

$$f(n,t,r,s)=|\{\{B_1,\dots,B_r\}\subseteq S_t\mid|\bigcap_{1\leqslant\ell\leqslant r}B_\ell|=s\}|.$$

The value of $f(n,t,r,s)$ is calculated in the next lemma and its proof appears in Appendix E.

Lemma 20. *For $1\leqslant r\leqslant|S_t|$, the value $f(n,t,r,s)$ satisfies*

$$f(n,t,r,s)=\binom{n}{s}\sum_{m=0}^{t-s}(-1)^m\binom{n-s}{m}\binom{\binom{n-s-m}{t-s-m}}{r}.$$

In the next claim, we list several combinatorial identities, which their proof is omitted as an exercise for the reader.

Claim 9.

(a) For all positive integer a it holds

$$\sum_{i=1}^{a}(-1)^{i+1}\binom{a}{i}=1.$$

(b) For all positive integers a and b such that $b \geqslant a$ it holds

$$\sum_{i=0}^{a}(-1)^{i}\binom{b}{i}=(-1)^{a}\binom{b-1}{a}.$$

(c) For all positive integers a, b and c such that $c \geqslant b \geqslant a$ it holds

$$\binom{c}{a}\binom{c-a}{b-a}=\binom{c}{b}\binom{b}{a}.$$

(d) For all positive integers a, b and c such that $c \geqslant b \geqslant a$ it holds

$$\binom{c-a-1}{b-a}=\binom{c-a}{b-a}\frac{c-b}{c-a}.$$

We are now ready to prove the main result of this section.

Theorem 21. *For all positive integer n, prime power q and even k, $t = k/2$, any optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$ satisfies*

$$q^{\binom{2t+1}{2}} \geqslant \sum_{s=0}^{t}(-1)^{t-s}q^{\binom{s+1}{2}}\binom{n}{t}\binom{t}{s}\frac{n-t}{n-s}.$$

Proof: Let $\mathcal{C}_{\mathcal{U}}$ be an optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$ and let $\mathbf{G}$ be its generator matrix. We will use the inclusion-exclusion principle to calculate the number of columns that are not in $\bigcup_{B\in S_t}\operatorname{span} V_B$, where V_B is defined in (11).

We denote by W the number of options to choose every column of the matrix $\mathbf{G}$, that is, $W = q^{\binom{k+1}{2}} = q^{\binom{2t+1}{2}}$. We will calculate $|\bigcup_{B\in S_t}\operatorname{span} V_B|$ by the inclusion-exclusion principle,

$$\begin{aligned}|\bigcup_{B\in S_t}\operatorname{span} V_B| &= \\ &=\sum_{r=1}^{|S_t|}(-1)^{r+1}\Big(\sum_{B_1,B_2,\ldots,B_r\in S_t}|\operatorname{span} V_{B_1}\cap\cdots\cap\operatorname{span} V_{B_r}|\Big).\end{aligned}$$

Next, we present the following calculations and explain each step afterwards.

$$\begin{aligned}
&|\bigcup_{B\in S_t}\operatorname{span} V_B| = \\
&=\sum_{r=1}^{|S_t|}(-1)^{r+1}\Big(\sum_{B_1,B_2,\ldots,B_r\in S_t}|\operatorname{span} V_{B_1}\cap\cdots\cap\operatorname{span} V_{B_r}|\Big)\\
&\overset{\text{(I)}}{=}\sum_{r=1}^{|S_t|}(-1)^{r+1}\Big(\sum_{B_1,B_2,\ldots,B_r\in S_t}|\operatorname{span}(V_{B_1}\cap\cdots\cap V_{B_r})|\Big)\\
&\overset{\text{(II)}}{=}\sum_{r=1}^{|S_t|}(-1)^{r+1}\Big(\sum_{\substack{B_1,B_2,\ldots,B_r\in S_t\\ |\bigcap_{1\leqslant i\leqslant r}B_i|=s}}q^{\binom{s+1}{2}}\Big)\\
&\overset{\text{(III)}}{=}\sum_{r=1}^{|S_t|}(-1)^{r+1}\Big(\sum_{s=0}^{t}q^{\binom{s+1}{2}}f(n,t,r,s)\Big)\\
&\overset{\text{(IV)}}{=}\sum_{r=1}^{|S_t|}(-1)^{r+1}\Big(\sum_{s=0}^{t}q^{\binom{s+1}{2}}\binom{n}{s}\Big(\sum_{m=0}^{t-s}(-1)^{m}\binom{n-s}{m}\Big(\binom{\binom{n-s-m}{t-s-m}}{r}\Big)\Big)\Big)\\
&=\sum_{s=0}^{t}q^{\binom{s+1}{2}}\binom{n}{s}\Big(\sum_{m=0}^{t-s}(-1)^{m}\binom{n-s}{m}\Big(\sum_{r=1}^{|S_t|}(-1)^{r+1}\binom{\binom{n-s-m}{t-s-m}}{r}\Big)\Big)\\
&\overset{\text{(V)}}{=}\sum_{s=0}^{t}q^{\binom{s+1}{2}}\binom{n}{s}\Big(\sum_{m=0}^{t-s}(-1)^{m}\binom{n-s}{m}\Big(\sum_{r=1}^{\binom{n-s-m}{t-s-m}}(-1)^{r+1}\binom{\binom{n-s-m}{t-s-m}}{r}\Big)\Big)\\
&\overset{\text{(VI)}}{=}\sum_{s=0}^{t}q^{\binom{s+1}{2}}\binom{n}{s}\Big(\sum_{m=0}^{t-s}(-1)^{m}\binom{n-s}{m}\Big)\\
&\overset{\text{(VII)}}{=}\sum_{s=0}^{t}q^{\binom{s+1}{2}}\binom{n}{s}(-1)^{t-s}\binom{n-s-1}{t-s}\\
&\overset{\text{(VIII)}}{=}\sum_{s=0}^{t}(-1)^{t-s}q^{\binom{s+1}{2}}\binom{n}{s}\binom{n-s}{t-s}\frac{n-t}{n-s}\\
&\overset{\text{(IX)}}{=}\sum_{s=0}^{t}(-1)^{t-s}q^{\binom{s+1}{2}}\binom{n}{t}\binom{t}{s}\frac{n-t}{n-s}.
\end{aligned}$$

Equality (I) holds since by Lemma 19, $|\big(\bigcap_{1\leqslant i\leqslant r}\operatorname{span} V_{B_i}\big)| = |\operatorname{span}\big(\bigcap_{1\leqslant j\leqslant r} V_{B_i}\big)|$. Equality (II) holds since for $0 \leqslant s \leqslant t$, if $|\bigcap_{1\leqslant i\leqslant r} B_i| = s$ then $|\operatorname{span}\big(\bigcap_{1\leqslant i\leqslant r} V_{B_i}\big)| = q^{\binom{s+1}{2}}$. Equality (III) holds since by definition the number of options such that $|\bigcap_{1\leqslant i\leqslant r} B_i| = s$ is $f(n,t,r,s)$. Equality (IV) holds by Lemma 20. Equality (V) holds since $|S_t| = \binom{n}{t} > \binom{n-s-m}{t-s-m}$, and for $r > \binom{n-s-m}{t-s-m}$, $\binom{\binom{n-s-m}{t-s-m}}{r}$ will be zero. Equality (VI), (VII), (VIII), (IX) holds by Claim 9(a), 9(b), 9(c), 9(d), respectively.

Denote by E, the number of columns of $\mathbb{F}_q^{\binom{k+1}{2}}$ that are not in $\bigcup_{B\in S_t}\operatorname{span} V_B$, that is, $E = W - |\bigcup_{B\in S_t}\operatorname{span} V_B|$. Therefore, the code $\mathcal{C}_{\mathcal{U}}$ exists only if E is not smaller than 0, or, $W \geqslant |\bigcup_{B\in S_t}\operatorname{span} V_B|$, that is,

$$q^{\binom{2t+1}{2}} \geqslant \sum_{s=0}^{t}(-1)^{t-s}q^{\binom{s+1}{2}}\binom{n}{t}\binom{t}{s}\frac{n-t}{n-s}.$$

■

For $k = 2$ $(t = 1)$ the bound from Theorem 21 states that

$$\begin{aligned}q^3 &\geqslant \sum_{s=0}^{1}(-1)^{1-s}q^{\binom{s+1}{2}}\binom{n}{1}\binom{1}{s}\frac{n-1}{n-s}\\ &= -n\cdot\frac{n-1}{n}+q\cdot n\cdot\frac{n-1}{n-1}\\ &= -n+1+q\cdot n = 1+(q-1)n,\end{aligned}$$

or, $q^3 - 1 \geqslant (q-1)n$, that is equivalent to

$$q^2+q+2>n,$$

which is the result of Theorem 18. For $q = 2$ we get that there is no binary code for $n > 7$. Similarly, it can be verified that for $k = 4$, there is no binary code for $n > 20$, and for $k = 6$ there is no binary code for $n > 69$.

In the next corollary, for fixed values of k and q, we find an upper bound on the value of n for the existence of optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$.

Corollary 22. *Let k be a fixed positive integer and let q be a prime power. For all n such that $\log_q(n) > \frac{3}{4}k + \log_q(k) + \frac{3}{2}$, there does not exist an optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$.*

Proof: We present the following calculations and explain each step afterwards. From Theorem 21 we get that for $t = k/2$ the following inequality holds.

$$\begin{aligned}
q^{\binom{2t+1}{2}} &\geqslant \sum_{s=0}^{t}(-1)^{t-s}q^{\binom{s+1}{2}}\binom{n}{t}\binom{t}{s}\frac{n-t}{n-s}\\
&= q^{\binom{t+1}{2}}\binom{n}{t} - \binom{n}{t}\sum_{s=0}^{t-1}(-1)^{t-s}q^{\binom{s+1}{2}}\binom{t}{s}\frac{n-t}{n-s}\\
&\overset{\text{(I)}}{\geqslant} q^{\binom{t+1}{2}}\binom{n}{t} - \binom{n}{t}\sum_{s=0}^{t-1}(-1)^{t-s}q^{\binom{s+1}{2}}\binom{t}{s}\\
&\overset{\text{(II)}}{\geqslant} q^{\binom{t+1}{2}}\binom{n}{t} - \binom{n}{t}\sum_{\substack{s\in[t]\\ (t-s)\text{ is even}}} q^{\binom{s+1}{2}}\binom{t}{s}\\
&\overset{\text{(III)}}{\geqslant} q^{\binom{t+1}{2}}\binom{n}{t} - q^{\binom{t}{2}}\binom{n}{t}\sum_{\substack{s\in[t]\\ (t-s)\text{ is even}}}\binom{t}{t-s}\\
&= q^{\binom{t+1}{2}}\binom{n}{t} - q^{\binom{t}{2}}\binom{n}{t}\sum_{i=0}^{\lfloor t/2\rfloor}\binom{t}{2i}\\
&\overset{\text{(IV)}}{=} q^{\binom{t+1}{2}}\binom{n}{t} - q^{\binom{t}{2}}\binom{n}{t}2^{t-1}\\
&\overset{\text{(V)}}{\geqslant} q^{\binom{t+1}{2}}\binom{n}{t} - \frac{1}{2}q^{\binom{t}{2}}\binom{n}{t}q^{t}\\
&= q^{\binom{t+1}{2}}\binom{n}{t} - \frac{1}{2}q^{\binom{t+1}{2}}\binom{n}{t}\\
&= \frac{1}{2}q^{\binom{t+1}{2}}\binom{n}{t}.
\end{aligned}$$

Inequality (I) holds since $0 \leqslant s \leqslant t$, so $\frac{n-t}{n-s} \leqslant 1$. Inequality (II) holds since we remove all the cases where $(-1)^{t-s}$ is negative. Inequality (III) holds since for all $s \in [t]$, $q^{\binom{s+1}{2}} \leqslant q^{\binom{t}{2}}$ and $\binom{t}{s} = \binom{t}{t-s}$. Equality (IV) holds based upon the formula $\sum_{i=0}^{\lfloor t/2\rfloor}\binom{t}{2i} = 2^{t-1}$. Inequality (V) holds since $2^t \leqslant q^t$.

Therefore, we deduce that for any optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$

$$q^{\binom{k+1}{2}} \geqslant \frac{1}{2}q^{\binom{k/2+1}{2}}\binom{n}{k/2}.$$

Since for positive integer a and fixed b, such that $b \leqslant a$, $\binom{a}{b} \geqslant \frac{(a-b)^b}{b!}$, we get that

$$\begin{aligned}
q^{\binom{k+1}{2}-\binom{k/2+1}{2}} &\geqslant \frac{1}{2}\binom{n}{k/2} \geqslant \frac{(n-k)^{k/2}}{2(k/2)!},\\
q^{\frac{3}{8}k^2+\frac{1}{4}k}2(k/2)! &\geqslant (n-k)^{k/2},\\
q^{\frac{3}{4}k+\frac{1}{2}}\sqrt[k/2]{2(k/2)!} &\geqslant n-k,\\
q^{\frac{3}{4}k+\frac{1}{2}}\sqrt[k/2]{2(k/2)!}+k &\geqslant n.
\end{aligned}$$

Since $\sqrt[k/2]{2(k/2)!} \leqslant 2\sqrt[k/2]{(k/2)!} \leqslant 2k/2$ we can write

$$\begin{aligned}
q^{\frac{3}{4}k+\frac{1}{2}}k + k &\geqslant n,\\
(q^{\frac{3}{4}k+\frac{1}{2}}+1)k &\geqslant n.
\end{aligned}$$

and

$$\log_q(q^{\frac{3}{4}k+\frac{1}{2}}+1)+\log_q(k) \geqslant \log_q(n).$$

Since for all $a > 1$, $\log_q(a)+1 \geqslant \log_q(a+1)$, we conclude that

$$\frac{3}{4}k+\frac{3}{2}+\log_q(k) \geqslant \log_q(n).$$

Therefore, for all n such that $\log_q(n) > \frac{3}{4}k + \log_q(k) + \frac{3}{2}$ an optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$ does not exist. ■

For odd k by the same method we can deduce that for all n satisfying the same inequality $\log_q(n) > \frac{3}{4}k + \log_q(k) + \frac{3}{2}$ an optimal undirected $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$ does not exist. Moreover, for directed graphs and any fixed k, for all n satisfying the inequality $\log_q(n) > \frac{3}{2}k + \log_q(k)+1$ an optimal directed $(n-k)$-node-erasure-correcting code over $\mathbb{F}_q$ does not exist.

VIII. Conclusion

In this paper we proposed a new construction of codes, called codes over graphs. We studied here complete undirected or directed graphs and the goal was to construct codes over graphs which are capable to correct the erasure of node failures. We built upon previous constructions of product codes and rank metric codes. The former set of codes provided us with optimal codes with linear field size and the latter was used for the construction of non-optimal binary codes. We were then aspired by the construction of EVENODD codes in order to construct optimal codes over graphs correcting two node failures over the binary field. Lastly, we studied upper bounds on the number of nodes of optimal codes.

Appendix A

Claim 1 *For all $h \in [n-1]$, $|S_h| = n-1$ and for all $m \in [n]$, $|D_m| = \frac{n+1}{2}$.*

Proof: The first part of the claim is readily verified. For $m \in [n]$, by the definition of D_m,

$$\{\langle v_k, v_\ell\rangle \mid k,\ell \in [n]\setminus\{n-2\}, \langle k+\ell\rangle_n = m\} \cap \{\langle v_{n-1}, v_{n-2}\rangle\} = \emptyset.$$

Note that,

$$\begin{aligned}
&|\{\langle v_k, v_\ell\rangle \mid k,\ell \in [n]\setminus\{n-2\}, \langle k+\ell\rangle_n = m\}|\\
=&|\{\langle v_k, v_{\langle m-k\rangle_n}\rangle \mid k, \langle m-k\rangle_n \in [n]\setminus\{n-2\}\}|\\
=&|\{\langle v_k, v_{\langle m-k\rangle_n}\rangle \mid k \in [n]\setminus\{n-2, \langle m+2\rangle_n\}\}|.
\end{aligned}$$

If $\langle m+2\rangle_n \neq n-2$ then k gets $n-2$ distinct values and there are $n-2$ options for edge $\langle v_k, v_{\langle m-k\rangle_n}\rangle$. For each of the $n-2$ options either $k \neq \langle m-k\rangle_n$ and we get each edge counted twice since $\langle v_k, v_{\langle m-k\rangle_n}\rangle = \langle v_{\langle m-k\rangle_n}, v_k\rangle$, or $k = \langle m-k\rangle_n$ and we get the self loop $\langle v_k, v_k\rangle$. Therefore,

$$|\{\langle v_k, v_{\langle m-k\rangle_n}\rangle \mid k \in [n] \setminus \{n-2, \langle m+2\rangle_n\}\}| = \frac{n-3}{2} + 1 = \frac{n-1}{2}.$$

If $\langle m+2\rangle_n = n-2$ then k gets $n-1$ distinct values and there are $n-1$ options for edge $\langle v_k, v_{\langle m-k\rangle_n}\rangle$. For each of the $n-1$ options if $k \neq \langle m-k\rangle_n$ we get each edge counted twice since $\langle v_k, v_{\langle m-k\rangle_n}\rangle = \langle v_{\langle m-k\rangle_n}, v_k\rangle$. Notice that there is no option for self loop $\langle v_k, v_k\rangle$ since it can be generated only for $k = n-2$. Therefore, in this case we also have that

$$|\{\langle v_k, v_{\langle m-k\rangle_n}\rangle \mid k \in [n] \setminus \{n-2\}\}| = \frac{n-1}{2}.$$

Therefore for all $m \in [n]$ we have that

$$|D_m| = |\{\langle v_k, v_\ell\rangle \mid k, \ell \in [n] \setminus \{n-2\}, \langle k+\ell\rangle_n = m\}| + |\{\langle v_{n-1}, v_{n-2}\rangle\}| = \frac{n-1}{2} + 1 = \frac{n+1}{2}.$$

■

APPENDIX B

Claim 2 *The sets S_h, D_m, F_t satisfy the following properties.*

(a) For all distinct $h, i \in [n-2]$, $S_h \cap F_i = \{\langle v_h, v_i\rangle\}$.
(b) For all $i \in [n-2]$, $S_{n-2} \cap F_i = \{\langle v_i, v_i\rangle\}$.
(c) For all pairwise distinct $i, j, h \in [n-2]$, $S_h \cap (F_i \cup F_j) = \{\langle v_h, v_i\rangle, \langle v_h, v_j\rangle\}$.
(d) For all distinct $i, j \in [n-2]$, $S_{n-2} \cap (F_i \cup F_j) = \{\langle v_i, v_i\rangle, \langle v_j, v_j\rangle\}$.
(e) For all $i \in [n-2]$, $D_{\langle i-2\rangle_n} \cap F_i = \emptyset$ and for $i = n-1$, $D_{\langle n-3\rangle_n} \cap F_{n-1} = \{\langle v_{n-2}, v_{n-1}\rangle\}$.
(f) For all $i \in [n-2]$, $s \in [n] \setminus \{\langle i-2\rangle_n\}$, $D_s \cap F_i = \{\langle v_{\langle s\text{-}i\rangle_n}, v_i\rangle\}$. For i such that $i = n-1$, $D_s \cap F_{n-1} = \{\langle v_{\langle s+1\rangle_n}, v_{n-1}\rangle, \langle v_{n-2}, v_{n-1}\rangle\}$
(g) For all distinct $i, j \in [n-2]$, $D_{\langle i+j\rangle_n} \cap F_j = \{\langle v_i, v_j\rangle\}$.
(h) For all distinct $i, j \in [n-2]$, $D_{\langle j-2\rangle_n} \cap (F_i \cup F_j) = \{\langle v_{\langle j-i-2\rangle_n}, v_i\rangle\}$
(i) For all distinct $i, j \in [n-2]$, $D_{\langle i+j\rangle_n} \cap (F_i \cup F_j) = \{\langle v_i, v_j\rangle\}$.
(j) For all $i \in [n-2]$, $D_{\langle i-2\rangle_n} \cap (F_i \cup F_{n-2}) = \{\langle v_{n-1}, v_{n-2}\rangle\}$.

Proof:

(a) Assume that there is an edge $\langle v_h, v_\ell\rangle \in S_h \cap F_i$. Since $\langle v_h, v_\ell\rangle \in F_i$ such that $h \neq i$, deduce that $\ell = i$. Therefore $S_h \cap F_i = \{\langle v_h, v_i\rangle\}$.

(b) Assume that there is an edge $\langle v_k, v_k\rangle \in S_{n-2} \cap F_i$, thus, $k = i$. Therefore $S_{n-2} \cap F_i = \{\langle v_i, v_i\rangle\}$.

(c) Using (a) deduce,

$$S_h \cap (F_i \cup F_j) = (S_h \cap F_i) \cup (S_h \cap F_j) = \{\langle v_h, v_i\rangle, \langle v_h, v_j\rangle\}$$

(d) Using (b) deduce,

$$S_{n-2} \cap (F_i \cup F_j) = (S_{n-2} \cap F_i) \cup (S_{n-2} \cap F_j) = \{\langle v_i, v_i\rangle, \langle v_j, v_j\rangle\}$$

(e) For $i < n-2$, $\langle v_{n-1}, v_{n-2}\rangle \notin F_i$, it is enough to show that

$$\{\langle v_k, v_\ell\rangle \mid k, \ell \in [n] \setminus \{n-2\}, \langle k+\ell\rangle_n = \langle i-2\rangle_n\} \cap \{\langle v_i, v_\ell\rangle \mid \ell \in [n]\} = \emptyset.$$

Assume that there is an edge $\langle v_k, v_\ell\rangle \in D_{\langle i-2\rangle_n} \cap F_i$, where $k, \ell \in [n] \setminus \{n-2\}$ and in particular $k, \ell \neq n-2$. According to the definition of the set F_i we have that $k = i$ or $\ell = i$. Without loss of generality assume that $k = i$ and according to the definition of the set $D_{\langle i-2\rangle_n}$ we have that $\langle i+\ell\rangle_n = \langle i-2\rangle_n$ and thus $\ell = n-2$, in contradiction. By the same concept, it is easy to verify that for $i = n-1$, the only edge in intersection of D_{n-3} and F_{n-1} is $\langle v_{n-1}, v_{n-2}\rangle$.

(f) Since $i < n-2$, $\langle v_{n-1}, v_{n-2}\rangle \notin F_i$, so it is enough to show that

$$\{\langle v_k, v_\ell\rangle \mid k, \ell \in [n] \setminus \{n-2\}, \langle k+\ell\rangle_n = s\} \cap \{\langle v_i, v_\ell\rangle \mid \ell \in [n]\} = \{\langle v_i, v_{\langle s\text{-}i\rangle_n}\rangle\}.$$

Assume that there is an edge $\langle v_k, v_\ell\rangle \in D_s \cap F_i$. According to the definition of the set F_i we have that $k = i$ or $\ell = i$. Without loss of generality assume that $k = i$. By definition of the set D_s we deduce that $\langle i+\ell\rangle_n = s$, so $\ell = \langle s\text{-}i\rangle_n$. In case where $i = n-1$, we simply add the $\langle v_{n-1}, v_{n-2}\rangle$ edge to the intersection between D_s and F_{n-1}, and the remaining proof is similar.

(g) Let $s = \langle i+j\rangle_n$. Since $j < n-2$, $s = \langle i+j\rangle_n \neq \langle i-2\rangle_n$, therefore according to (f) we deduce that $D_{\langle i+j\rangle_n} \cap F_i = \{\langle v_i, v_j\rangle\}$.

(h) Let $s = \langle j-2\rangle_n$. Since $i \neq j$, $s = \langle j-2\rangle_n \neq \langle i-2\rangle_n$, therefore according to (f) we deduce that $D_{\langle j-2\rangle_n} \cap F_i = \{\langle v_{\langle j-i-2\rangle_n}, v_i\rangle\}$, and since $j \in [n-2]$, by (e) deduce that $D_{\langle j-2\rangle_n} \cap F_j = \emptyset$. Therefore:

$$D_{\langle j-2\rangle_n} \cap (F_i \cup F_j) = (D_{\langle j-2\rangle_n} \cap F_i) \cup (D_{\langle j-2\rangle_n} \cap F_j) = \{\langle v_{\langle j-i-2\rangle_n}, v_i\rangle\}.$$

(i) Let $s = \langle i+j\rangle_n$, therefore,

$$D_{\langle i+j\rangle_n} \cap (F_i \cup F_j) = (D_{\langle i+j\rangle_n} \cap F_i) \cup (D_{\langle i+j\rangle_n} \cap F_j) = \{\langle v_i, v_j\rangle\} \cup \{\langle v_j, v_i\rangle\} = \{\langle v_i, v_j\rangle\}$$

where equality holds according to (g).

(j) Since $i \in [n-2]$, by (e) we get that $D_{\langle i-2\rangle_n} \cap F_i = \emptyset$. Since the set $D_{\langle i-2\rangle_n}$ has only one edge connecting the node v_{n-2}, we get that $D_{\langle i-2\rangle_n} \cap F_{n-2} = \{\langle v_{n-1}, v_{n-2}\rangle\}$, and therefore:

$$D_{\langle i-2\rangle_n} \cap (F_i \cup F_{n-2}) = (D_{\langle i-2\rangle_n} \cap F_i) \cup (D_{\langle i-2\rangle_n} \cap F_{n-2}) = \{\langle v_{n-1}, v_{n-2}\rangle\}.$$

■

APPENDIX C

Claim 3 *The following properties hold:*

(a) $x \neq y$ and $x + y = n-2$.
(b) $s_1^{(x)} = \widetilde{s}_1^{(y)} = n-1$.
(c) $i, j \in A$ or $i, j \in B$ but not in both.
(d) $n-2 \notin A \cup B$.
(e) $A \cap B = \{n-1\}$.
(f) $|A| = x+1$ and $|B| = y+1$.
(g) $s_1^{(0)} \neq i$ and $\widetilde{s}_1^{(0)} \neq j$.
(h) For $0 < t \leqslant x$, $s_1^{(t-1)} = j$ if and only if $s_1^{(t)} = i$.

(i) For $0 < t \leqslant x$, $s_2^{(t)} \notin \{\langle i-2\rangle_n, \langle j-2\rangle_n\}$.

Proof: First we remind that n is a prime number.

(a) Since $x = \langle -1-d^{-1}\rangle_n$ and $y = \langle -1+d^{-1}\rangle_n$, we get

$$x - y = \langle -1-d^{-1}\rangle_n - \langle -1+d^{-1}\rangle_n = \langle -2d^{-1}\rangle_n.$$

Since n is a prime number, we deduce that $\langle -2d^{-1}\rangle_n \neq 0$ and so $x \neq y$. By definition of x and y,

$$\langle x+y\rangle_n = \langle -1-d^{-1}\rangle_n + \langle -1+d^{-1}\rangle_n = \langle -2\rangle_n.$$

Since $x, y \in [n]$ and $x \neq y$ we conclude that $x + y = n - 2$.

(b) According to the definition of $s_1^{(t)}$ we get for $t = x$ that

$$s_1^{(x)} = \langle -d(x+1)-2\rangle_n = \langle -d(-d^{-1})-2\rangle_n = \langle 1-2\rangle_n = n-1.$$

The proof that $\widetilde{s}_1^{(y)} = n-1$ is identical.

(c) Assume that $j \in A$. Since $j \neq n-1 = s_1^{(x)}$, there exists $t_1 < x$ such that $s_1^{(t_1)} = \langle -d(t_1+1)-2\rangle_n = j$. Hence,

$$s_1^{(t_1+1)} = \langle s_1^{(t_1)} - d\rangle_n = \langle j-(j-i)\rangle_n = i$$

and we get that $i \in A$. The proof that if $i \in A$ then $j \in A$ is similar.
Assume again that $j \in A$, and on contrary that $i \in B$, Since $i \neq n-1 = \widetilde{s}_1^{(y)}$, there exists $t_2 < y$ such that $\widetilde{s}_1^{(t_2)} = i$, that is

$$\langle d(t_2+1)-2\rangle_n = i.$$

Therefore

$$d = \text{j-i} = \langle -d(t_1+1)-2\rangle_n - \langle d(t_2+1)-2\rangle_n,$$

and

$$\langle d(t_1+t_2+3)\rangle_n = 0$$

which leads to a contradiction since $t_1+t_2+3 \leqslant x-1+y-1+3 = n-1$. Similarly we prove that if $i \notin B$ then $j \in A$.

(d) We will prove without loss of generality that $n-2 \notin A$. Assume in contrary that $n-2 \in A$, then there exists $0 \leqslant t \leqslant x$ such that

$$s_1^{(t)} = \langle -d(t+1)-2\rangle_n = n-2.$$

Therefore, $\langle d(t+1)\rangle_n = 0$, which leads to a contradiction.

(e) By (b), $n-1 \in A \cap B$. Assume on contrary that exists $h \neq n-1$ such that $h \in A \cap B$. Since $h \neq n-1 = s_1^{(x)} = \widetilde{s}_1^{(y)}$, there exist $t_1 < x, t_2 < y$ such that

$$h = \langle -d(t_1+1)-2\rangle_n = \langle d(t_2+1)-2\rangle_n.$$

Hence we get

$$\langle d(t_1+t_2+2)\rangle_n = 0,$$

and again we get a contradiction.

(f) Assume that $|A| < x+1$. Therefore there are $0 \leqslant t_2 < t_1 \leqslant x$ such that $s_1^{(t_1)} = s_1^{(t_2)}$. By definition of $s_1^{(t_1)}, s_1^{(t_2)}$ we deduce,

$$\langle -d(t_1+1)-2\rangle_n = \langle -d(t_2+1)-2\rangle_n.$$

Hence we get

$$\langle d(t_1-t_2)\rangle_n = 0,$$

and since $0 < t_1 - t_2 \leqslant x \leqslant n-2$ we get a contradiction. The $|B| = y+1$ is proved similarly.

(g) Assume that $s_1^{(0)} = i$, therefore

$$\langle -d-2\rangle_n = \langle i-j-2\rangle_n = i,$$
$$\langle n-j-2\rangle_n = 0,$$
$$n-2 = j,$$

and that is a contradiction. The $\widetilde{s}_1^{(0)} \neq j$ is proved similarly.

(h) If $s_1^{(t-1)} = j$ then,

$$s_1^{(t)} = s_1^{(t-1)} - d = j-(j-i) = i.$$

If $s_1^{(t)} = i$ then,

$$s_1^{(t-1)} = s_1^{(t)} + d = i + (j-i) = j.$$

(i) If $s_2^{(t)} = \langle j-2\rangle_n$ then

$$s_1^{(t)} = \langle s_2^{(t)} - j\rangle_n = \langle (j-2)-j\rangle_n = n-2,$$

and we know that $s_1^{(t)} \neq n-2$ by (d). If $0 < t \leqslant x$ and $s_2^{(t)} = \langle i-2\rangle_n$ then

$$s_1^{(t)} = \langle s_2^{(t)} - j\rangle_n = \langle (i-2)-j\rangle_n = \langle -d-2\rangle_n = s_1^{(0)},$$

but since $|A| = x+1$, $s_1^{(t)} = s_1^{(0)}$ only for $t = 0$ and that is a contradiction.

■

APPENDIX D

Claim 4 *The following properties hold:*

(a) $V_1 \cap V_2 = \emptyset$.
(b) $|V| = 2n-4$.
(c) $\langle v_i, v_j\rangle \notin V$.
(d) $\langle v_{n-2}, v_i\rangle \notin V$ *and* $\langle v_{n-2}, v_j\rangle \notin V$.

Proof:

(a) According to Claim 3(e), $A \cap B = \{n-1\}$ and we get

$$\{\langle v_{s_1}, v_i\rangle, \langle v_{s_1}, v_j\rangle : s_1 \in A \setminus \{n-1\}\} \cap \{\langle v_{\widetilde{s}_1}, v_j\rangle, \langle v_{\widetilde{s}_1}, v_i\rangle : \widetilde{s}_1 \in B \setminus \{n-1\}\} = \emptyset.$$

By the definition of V_1 and V_2,

$$\{\langle v_{n-1}, v_j\rangle, \langle v_i, v_i\rangle, \langle v_j, v_j\rangle\} \subseteq V_1 \setminus V_2,$$

and $\langle v_{n-1}, v_i\rangle \in V_2 \setminus V_1$, therefore $V_1 \cap V_2 = \emptyset$.

(b) Since $|A| = x+1$ and since $i, j \in A$ we get that,

$$|\{\langle v_{s_1}, v_i\rangle, \langle v_{s_1}, v_j\rangle : s_1 \in A \setminus \{n-1\}\}| = 2(|A|-1)-2 = 2(x+1-1)-2 = 2x-2,$$

thus, $|V_1| = 2x - 1$. Similarly, since $|B| = y + 1$ and since $i, j \notin B$ we get that,

$$\begin{aligned}&|\{\langle v_{\widetilde{s}_1}, v_j\rangle, \langle v_{\widetilde{s}_1}, v_i\rangle : \widetilde{s}_1 \in B \setminus \{n-1\}\}|\\ &= 2(|B|-1) = 2(y+1-1) = 2y,\end{aligned}$$

thus, $|V_2| = 2y + 1$. By (a), $V_1 \cap V_2 = \emptyset$ we deduce,

$$\begin{aligned}|V| &= |V_1| \cup |V_2| = |V_1| + |V_2|\\ &= 2x - 1 + 2y + 1 = 2(x+y) = 2(n-2) = 2n-4.\end{aligned}$$

(c) According to the definition of V_1 and V_2, $\langle v_i, v_j\rangle \notin V_1$ and $\langle v_i, v_j\rangle \notin V_2$.

(d) By Claim 3(d), $n-2 \notin A \cup B$, therefore by the definition of V_1 and V_2, $\langle v_{n-2}, v_j\rangle, \langle v_{n-2}, v_i\rangle \notin V_1$ and $\langle v_{n-2}, v_j\rangle, \langle v_{n-2}, v_i\rangle \notin V_2$.

■

APPENDIX E

Lemma 20 *For $1 \leqslant r \leqslant |S_t|$, the value $f(n,t,r,s)$ satisfies*

$$f(n,t,r,s) = \binom{n}{s} \sum_{m=0}^{t-s} (-1)^m \binom{n-s}{m} \binom{\binom{n-s-m}{t-s-m}}{r}.$$

Proof: Note that $f(n,t,r,s)$ is the cardinality of the set

$$H = \{\{B_1,\ldots,B_r\} \subseteq S_t \mid |\bigcap_{1\leqslant \ell \leqslant r} B_\ell| = s\}.$$

For a fixed set $B \subseteq [n]$ (e.g., $B = [s]$), we denote by H_B the set

$$H_B = \{\{B_1,\ldots,B_r\} \subseteq S_t \mid \bigcap_{1\leqslant \ell \leqslant r} B_\ell = B\}.$$

Since there are $\binom{n}{s}$ options to choose the set B, we deduce that

$$|H| = \sum_{B\subseteq[n]} |H_B| = \binom{n}{s} |H_{[s]}|.$$

For the remainder of the proof, we find the side of the set $H_{[s]}$.

We denote the set

$$W = \{\{B_1,\ldots,B_r\} \subseteq S_t \mid [s] \subseteq \bigcap_{1\leqslant \ell \leqslant r} B_\ell\},$$

where $|W| = \binom{\binom{n-s}{t-s}}{r}$. For $i \in ([n] \setminus [s])$ we also define A_i as follows

$$A_i = \{\{B_1,\ldots,B_r\} \in W |, i \in \bigcap_{1\leqslant \ell \leqslant r} B_\ell\},$$

where it holds that $|H_{[s]}| = |W| - |\bigcup_{i\in([n]\setminus[s])} A_i|$. Therefore, we calculate $|\bigcup_{i\in([n]\setminus[s])} A_i|$ by using the inclusion-exclusion principle

$$|\bigcup_{i\in([n]\setminus[s])} A_i| = \sum_{m=1}^{n-s} (-1)^{m+1} \Big(\sum_{s\leqslant i_1 < \cdots < i_m \leqslant n-1} |A_{i_1} \cap \cdots \cap A_{i_r}| \Big).$$

For all $1 \leqslant m \leqslant n-s$ and $0 \leqslant i_1 < \cdots < i_m \leqslant n-1$, it holds

$$|A_{i_1} \cap \cdots \cap A_{i_m}| = \binom{\binom{n-s-m}{t-s-m}}{r},$$

since the intersection $A_{i_1} \cap \cdots \cap A_{i_m}$ includes at least the set $[s]\cup\{i_1,\ldots,i_m\}$. Notice that for $m > t-s$, $|A_{i_1} \cap \cdots \cap A_{i_m}| = 0$ so we can take only the cases where $1 \leqslant m \leqslant t-s$. Therefore we can write

$$\begin{aligned}|\bigcup_{i\in([n]\setminus[s])} A_i| &= \sum_{m=1}^{n-s} (-1)^{m+1} \Big(\sum_{s\leqslant i_1 < \cdots < i_m \leqslant n-1} |A_{i_1} \cap \cdots \cap A_{i_r}| \Big)\\ &= \sum_{m=1}^{t-s} (-1)^{m+1} \binom{n-s}{m} \binom{\binom{n-s-m}{t-s-m}}{r}.\end{aligned}$$

Finally, we conclude that

$$|H_{[s]}| = |W| - |\bigcup_{i\in([n]\setminus[s])} A_i| = \sum_{m=0}^{t-s} (-1)^m \binom{n-s}{m} \binom{\binom{n-s-m}{t-s-m}}{r},$$

and since $f(n,t,r,s) = |H|$ we get

$$f(n,t,r,s) = \binom{n}{s} |H_{[s]}| = \binom{n}{s} \sum_{m=0}^{t-s} (-1)^m \binom{n-s}{m} \binom{\binom{n-s-m}{t-s-m}}{r}.$$

■

ACKNOWLEDGMENTS

The authors would like to thank Jehoshua Bruck for valuable discussions and Ron M. Roth for his contribution to the results in Section VII and sharing reference [16] with them.

REFERENCES

[1] N. Abramson, "Cascade decoding of cyclic product codes," *IEEE Trans. Commun. Technol.*, vol. 16, no. 3, pp. 398–402, Jun. 1968.

[2] M. Blaum, J. Brady, J. Bruck, and J. Menon, "EVENODD: An efficient scheme for tolerating double disk failures in RAID architectures," *IEEE Trans. Comput.*, vol. 44, no. 2, pp. 192–202, Feb. 1995.

[3] M. Blaum, J. Bruck, and A. Vardy, "MDS array codes with independent parity symbols," *IEEE Trans. Inf. Theory*, vol. 42, no. 2, pp. 529–542, Mar. 1996.

[4] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[5] P. Elias, "Error-free coding," *IRE Trans. IRE Prof. Group Inf. Theory*, vol. 4, no. 4, pp. 29–37, Sep. 1954.

[6] E. E. Gad, R. Mateescu, F. Blagojevic, C. Guyot, and Z. Bandic, "Repair-optimal MDS array codes over GF(2)," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, Jul. 2013, pp. 887–891.

[7] J. Hopfield, *Neurocomputing: Foundations of Research*. Cambridge, MA, USA: MIT Press, 1988, pp. 457–464.

[8] C. Huang and L. Xu, "STAR : An efficient coding scheme for correcting triple storage node failures," *IEEE Trans. Comput.*, vol. 57, no. 7, pp. 889–901, Jul. 2008.

[9] P. Corbett *et al.*, "Row-diagonal parity for double disk failure correction," in *Proc. 3rd USENIX Symp. File Storage Technol.*, San Francisco, CA, USA, Apr. 2004, pp. 1–14.

[10] S. K. Mohammed, E. Viterbo, Y. Hong, and A. Chockalingam, "X-codes: A low complexity full-rate high-diversity achieving precoder for TDD MIMO systems," in *Proc. IEEE Int. Conf. Commun.*, Cape Town, South Africa, May 2010, pp. 1–5.

[11] H. D. Pfister, S. K. Emmadi, and K. Narayanan, "Symmetric product codes," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2015, pp. 282–290.

[12] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," *IEEE Trans. Inf. Theory*, vol. 57, no. 8, pp. 5227–5239, Aug. 2011.

[13] N. Raviv, N. Silberstein, and T. Etzion, "Constructions of high-rate minimum storage regenerating codes over small fields," *IEEE Trans. Inf. Theory*, vol. 63, no. 4, pp. 2015–2038, Apr. 2017.

[14] R. M. Roth, “Maximum-rank array codes and their application to criss-cross error correction,” *IEEE Trans. Inf. Theory*, vol. 37, no. 2, pp. 328–336, Mar. 1991.

[15] B. Sasidharan, M. Vajha, and P. V. Kumar. (Sep. 2016). “An explicit, coupled-layer construction of a high-rate MSR code with low sub-packetization level, small field size and all-node repair.” [Online]. Available: https://arxiv.org/abs/1607.07335

[16] K.-U. Schmidt, “Symmetric bilinear forms over finite fields of even characteristic,” *J. Combinat. Theory, A*, vol. 117, no. 8, pp. 1011–1026, May 2010.

[17] K.-U. Schmidt, “Symmetric bilinear forms over finite fields with applications to coding theory,” *J. Algebraic Combinatorics*, vol. 42, no. 2, pp. 635–670, Sep. 2015.

[18] I. Tamo, Z. Wang, and J. Bruck, “Zigzag codes: MDS array codes with optimal rebuilding,” *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1597–1616, Mar. 2013.

[19] L. Xu, V. Bohossian, J. Bruck, and D. G. Wagner, “Low-density MDS codes and factors of complete graphs,” *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 1817–1836, Sep. 1999.

[20] E. Yaakobi and J. Bruck, “On the uncertainty of information retrieval in associative memories,” in *Proc. IEEE Int. Symp. Inf. Theory*, Cambridge, MA, USA, vol. 45, no. 6, Jul. 2012, pp. 106–110.

[21] M. Ye and A. Barg, “Explicit constructions of high-rate MDS array codes with optimal repair bandwidth,” *IEEE Trans. Inf. Theory*, vol. 63, no. 4, pp. 2001–2014, Apr. 2017.

[22] M. Ye and A. Barg, “Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization,” *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6307–6317, Oct. 2017.

[23] L. Yohananov and E. Yaakobi, “Codes for graph erasures,” in *Proc. IEEE Int. Symp. Inf. Theory*, Aachen, Germany, Jul. 2017, pp. 844–848.

[24] L. Yohananov and E. Yaakobi, “Codes for erasures over directed graphs,” in *Proc. IEEE Inf. Theory Workshop*, Kaohsiung, Taiwan, Nov. 2017, pp. 116–120.

Lev Yohananov is a Ph.D. student in the Computer Science Department at the Technion—Israel Institute of Technology.

He received the B.Sc. and M.Sc. degrees in computer science from the Technion—Israel Institute of Technology, Haifa, Israel, in 2016 and 2018, respectively. His research interests include algebraic error correction coding, coding on graphs, and combinatorics.

Eitan Yaakobi (S’07–M’12–SM’17) is an Assistant Professor at the Computer Science Department at the Technion—Israel Institute of Technology. He received the B.A. degrees in computer science and mathematics, and the M.Sc. degree in computer science from the Technion—Israel Institute of Technology, Haifa, Israel, in 2005 and 2007, respectively, and the Ph.D. degree in electrical engineering from the University of California, San Diego, in 2011. Between 2011–2013, he was a postdoctoral researcher in the Department of Electrical Engineering at the California Institute of Technology. His research interests include information and coding theory with applications to non-volatile memories, associative memories, data storage and retrieval, and voting theory. He received the Marconi Society Young Scholar in 2009 and the Intel Ph.D. Fellowship in 2010–2011.