Balanced XOR-ed Coding

Katina Kralevska, Danilo Gligoroski, Harald Øverby

Balanced XOR-ed Coding

Katina Kralevska, Danilo Gligoroski, and Harald Øverby

Department of Telematics, Faculty of Information Technology, Mathematics and Electrical Engineering, Norwegian University of Science and Technology, Trondheim, Norway,
{katinak, danilog, haraldov}@item.ntnu.no

Abstract. This paper concerns with the construction of codes over $GF(2)$ which reach the max-flow for single source multicast acyclic networks with delay. The coding is always a bitwise XOR of packets with equal lengths, and is based on highly symmetrical and balanced designs. For certain setups and parameters, our approach offers additional plausible security properties: an adversary needs to eavesdrop at least max-flow links in order to decode at least one original packet.

Keywords: XOR coding, $GF(2)$, Latin squares, Latin rectangles

1 Introduction

Encoding and decoding over $GF(2)$ (GF stands for Galois field with 2 elements: 0 and 1) is more energy efficient than encoding and decoding in any other larger field. Recent studies concerning several new techniques in network coding [1] (Linear Network Coding (LNC) [12, 10] and Random Linear Network Coding (RLNC) [6]) confirmed that encoding and decoding over $GF(2)$ are up to two orders of magnitude less energy demanding and up to one order of magnitude faster than the encoding/decoding operations in larger fields [18, 14, 20].

The high computational complexity of packet encoding and decoding over large finite fields and its high energy cost which makes it unsuitable for practical implementation are the main motivation to seek for coding techniques only with XOR operations. The first theoretical work was done by Riis in [16] who showed that every solvable multicast network has a linear solution over $GF(2)$. Afterwards, XOR coding in wireless networks was presented in [9], where the main rule is that a node can XOR n packets together only if the next hop has all $n-1$ packets. A more general network coding problem which is called index coding is considered in [17, 15]. In [15] the authors address the coding problem by proposing coding over $GF(2)$. The encoding scheme is based on bitwise XORing by adding redundant bits, and the decoding scheme is based on a simple but bit after bit sequential back substitution method.

The main contribution of our work is a construction of codes over $GF(2)$ by using combinatorial designs (Latin squares and Latin rectangles) [4]. Its lower computation and energy cost makes it suitable for practical implementation on devices with limited processing and energy capacity like mobile phones and wireless sensors. We will illustrate the construction of codes by the following simple example.

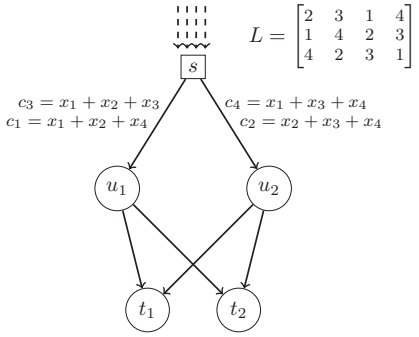

Fig. 1. An example of balanced XOR coding where the source sends combinations of source packets (combined as the column of the Latin rectangle). The intermediate nodes just forward the data to the sink nodes.

Example 1. We use the following strategy (Fig. 1): the source s performs bitwise XOR of packets with equal length based on the incidence matrix of a Latin rectangle L. Each column of L represents a combination of source packets x_i, $i = 1, \ldots, 4$, in a coded packet c_i, $i = 1, \ldots, 4$. In the first phase, the packets c_1 and c_2 are sent, and in the second phase, the packets c_3 and c_4 are sent. The intermediate nodes u_1 and u_2 forward the coded packets to the sink nodes t_1 and t_2 which decode the packets by using the inverse matrix of the incidence matrix of L. The sink nodes need only to know the combination of source packets in each received packet. Note that the max-flow in the network is achieved.

Routinely as in other coding approaches, this information is included in the header of each coded packet. Since in this paper we use diversity coding performed just by the source nodes, there is no need for updating the coefficients in the header at each intermediate node. The length of the prepended header vector is negligible compared to the length of the packet.

The construction of our codes was not motivated by security issues, therefore the security is not the main goal in this paper. However, it turns out that for certain setups and parameters, our approach offers additional plausible security properties. The plausible security properties that accompany our approach are not based on hard mathematical problems in modern cryptology (for example factoring of large integers or discrete logarithm problems or on the Shamir's secret sharing algorithm). We show that if an eavesdropper wants to reconstruct at least one original packet, then the number of eavesdropped links should be equal to the max-flow of the network. Bhattad and al. [2] make similar observations when network coding is implemented so that a weekly secure network coding is achieved.

The rest of the paper is organized as follows: Section 2 presents the notation and the mathematical background that are used in the following sections. The construction of codes is presented in Section 3. Section 4 illustrates the security features of our approach. Sections 5 concludes the paper.

2 Notation and Mathematical Background

We define a communication network as a tuple $N = (V, E, S, T)$ that consists of:

- a finite directed acyclic multigraph $G = (V, E)$ where V is the set of vertices and E is the set of edges,
- a set $S \subset V$ of sources,
- a set $T \subset V$ of sink nodes.

Assume that vertex $s \in S$ sends n source packets to vertex $t \in T$ over disjoint paths. A minimal cut separating s and t is a cut of the smallest cardinality denoted as mincut(s, t). The packets are sent in several time slots, i.e., phases denoted as p. The maximum number of packets that can be sent in a phase from s to t is denoted as maxflow(t). The Max-Flow Min-Cut Theorem [11] indicates that mincut(s, t) = maxflow(t). The multicast capacity, i.e., the maximum rate at which s can transfer information to the sink nodes, cannot exceed the capacity of any cut separating s from the sink nodes. A network is solvable when the sink nodes are able to deduce the original packets with decoding operations. If the network is solvable with linear operations we say that the network is linearly solvable.

2.1 XOR-ed coding

First we recall that in [16], Riis showed that every solvable multicast network has a linear solution over $GF(2)$ in some vector dimension. The essence of his proof relies on the fact that any two finite fields with the same cardinality are isomorphic. Thus, instead of working in a finite field $GF(2^n)$ for which the conditions of the linear-code multicast (LCM) theorem [12, Th. 5.1] are met, he showed that it is possible to work in the isomorphic vector space $GF(2)^n$ that is an extension field over the prime field $GF(2)$. We formalize the work in the vector space $GF(2)^n$ with the following:

Definition 1. *A XOR-ed coding is a coding that is realized exclusively by bitwise XOR operations between packets with equal length. Hence, it is a parallel bitwise linear transformation of n source bits $x = (x_1, \ldots, x_n)$ by a $n \times n$ nonsingular matrix K, i.e., $y = K \cdot x$.*

In [16] it was also shown that there are simple network topologies where encoding in $GF(2)$ cannot reach the network capacity with the original bandwidth or by sending data in just one phase. However, it was shown that the network capacity by XOR-ed coding can be achieved either by increasing the bandwidth or the number of phases so that they match the dimension of the extended vector space $GF(2)^n$. In this paper we take the approach to send data in several phases p instead of increasing the bandwidth.

Theorem 1. *For any linearly solvable network topology with maxflow$(t_1) > 1$, the sufficient condition for a single sink t_1 to reach its capacity in each of p phases by XOR-ed coding is to receive n linearly independent packets $x = (x_1, \ldots, x_n)$, where $n = p \times$ maxflow(t_1).*

Proof. Assume that the network topology is linearly solvable. That means there exists a vector space $GF(2)^n$ where we can encode every n source bits with a bijective function K, i.e., $y = K \cdot x$. Having in mind that the source s succeeds to send n encoded packets to t_1 in p phases, and the max-flow in the network is $\text{maxflow}(t_1) > 1$, we have that $n = p \times \text{maxflow}(t_1)$ and the sink t_1 receives n packets after p phases via $\text{maxflow}(t_1)$ disjoint paths. In order to have a successful recovery of the initial n packets, the received packets should be linearly independent.

Based on Theorem 1 we can prove the following:

Theorem 2. *For any linearly solvable network topology and for any two sinks $T = \{t_1, t_2\}$ that have maxflow$(t) =$ maxflow$(t_1) =$ maxflow(t_2), there always exists a XOR-ed coding for $n = p \times$ maxflow(t) packets that achieves the multicast capacity in each of p phases.*

Proof. For the sink t_1 we apply Theorem 1 and find one XOR-ed coding that achieves the capacity in each of p phases. Let us denote by $U_1 = \{u_{1,i}|$ there is an edge $(u_{1,i}, t_1) \in E\}$ the nodes that are directly connected and send packets to the sink node t_1. We have that $|U_1| = \text{maxflow}(t)$, and the set of n packets is partitioned in $\text{maxflow}(t)$ disjoint subsets $Y_{1,1}, \ldots, Y_{1,\text{maxflow}(t)}$ each of them having p packets. The subset $Y_{1,i}$ comes from the node u_i, $i = 1, \ldots, \text{maxflow}(t)$.

The set $U_2 = \{u_{2,i}|\text{there is an edge}(u_{2,i}, t_2) \in E\}$ is a set of nodes that are directly connected and send packets to the sink node t_2. We denote the intersection between the sets of nodes U_1 and U_2 as $U_{1,2} = U_1 \bigcap U_2$. The following three situations are considered:

1. There are no mutual nodes that send packets to both sinks t_1 and t_2, i.e., $U_{1,2} = \emptyset$. In that case find one partition of the set of n packets in $\text{maxflow}(t)$ disjoint subsets $\Gamma_1 = \{Y_{2,1} \ldots, Y_{2,\text{maxflow}(t)}\}$ each of them having p packets. The sets of packets $Y_{2,j}$ are delivered to the sink t_2 via the node $u_{2,j}$, $j = 1, \ldots, \text{maxflow}(t)$. The multicast capacity for the sink t_2 is achieved in each of p phases.
2. There are nodes that send packets to both sinks t_1 and t_2, i.e., $U_{1,2} = \{u_{(1,2)_{\nu_1}}, \ldots, u_{(1,2)_{\nu_k}}\}$. Denote the nodes that are in $U_2 \backslash U_1 = \{u_{2_{\nu_1}}, \ldots$ $\ldots, u_{2_{\nu_{\text{maxflow}(t)-k}}}\}$. In that case, the sink t_2 receives from the nodes in $U_{1,2}$ the same packets that are delivered to the sink t_1. The number of the remaining packets that have to be delivered to t_2 is exactly $p \times (\text{maxflow}(t) - k)$. Find one partition of $\text{maxflow}(t) - k$ disjoint subsets $\Gamma_2 = \{Y_{2,1} \ldots, Y_{2,\text{maxflow}(t)-k}\}$ each of them having p packets. The sets of packets $Y_{2,j}$ are delivered to the sink t_2 via the node u_{2,ν_j}, $j = 1, \ldots, \text{maxflow}(t) - k$. The multicast capacity for the sink t_2 is achieved in each of p phases.
3. All the nodes that send packets to the sink t_1, send packets to the sink t_2 as well, i.e., $U_{1,2} = U_1 \bigcap U_2 = U_1$. In that case, the sink t_2 receives from the nodes in $U_{1,2}$ the same packets that are delivered to the sink t_1. The multicast capacity for the sink t_2 is achieved in each of p phases.

Note that the proof of Theorem 2 is similar to the work by Jaggi et al. [8] where they discuss a construction of general codes using simple algorithms.

As a consequence of Theorems 1 and 2 we can post the following:

Theorem 3. *For any linearly solvable network topology and for any set of N sinks $T = \{t_1, \ldots, t_N\}$ that have $maxflow(t) = maxflow(t_1) = \ldots = maxflow(t_N)$, there always exists a XOR-ed coding for $n = p \times maxflow(t)$ packets that achieves the multicast capacity in each of p phases.*

Proof. (*Sketch*) First, we recall the construction of generic linear codes presented in the LCM theorem in [12, Th. 5.1]. Second, we use the transformation to equivalent codes over $GF(2)^n$ as it was shown in [16]. Then, the proof is a straightforward application of the mathematical induction by the number of sinks N. Let us suppose that the claim of the theorem is correct for $N-1$ sinks. By adding a new N-th sink we consider again three possible situations as in Theorem 2.

3 Construction of XOR-ed Coding

In this section we describe the construction of codes over $GF(2)$. Instead of working with completely random binary matrices, in the remaining part of this paper we work with nonsingular binary matrices that have some specific structure related to randomly generated Latin square or Latin rectangle. We do not reduce the space of possible random linear network encoding schemes, since the number of Latin squares and Latin rectangles of order n increases proportionally with factorial of n. Therefore, in our approach we have virtually an endless repository of encoding schemes that have the benefits from both worlds: they are randomly generated, but they have a certain structure and offer plausible security properties.

In order to introduce our approach, we briefly use several definitions that the reader can find in [19] and [3].

Definition 2. *A Latin square of order n with entries from an n-set X is an $n \times n$ array L in which every cell contains an element of X such that every row of L is a permutation of X and every column of L is a permutation of X.*

Definition 3. *A $k \times n$ Latin rectangle is a $k \times n$ array (where $k \leq n$) in which each cell contains a single symbol from an n-set X, such that each symbol occurs exactly once in each row and at most once in each column.*

For generating a Latin square, one can always start with a permutation of n elements that is a trivial $1 \times n$ Latin rectangle and can use the old Hall's marriage theorem [5] to construct new rows until the whole Latin square is completed. However, this approach does not guarantee that the generated Latin squares are chosen uniformly at random. In order to generate Latin squares of order n that are chosen uniformly at random we use the algorithm of Jacobsen and Matthews [7]. Further, in our approach we sometimes split the Latin square into two Latin

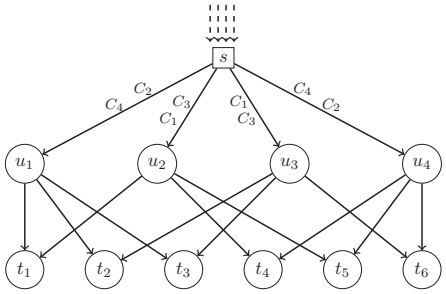

Fig. 2. A 4-dimensional binary linear multicast in a single source multicast network with delay

rectangles (upper and lower), and work with the algebraic objects (matrices or block designs) that are related to either the upper or the lower Latin rectangle.

As a convention, throughout this paper, the number of packets n that are sent from the source is equal to the number of columns in the Latin square or Latin rectangle.

Example 2. As shown in Fig. 2, we assume that the source wants to send four packets $x_1, \ldots, x_4$ to the sink nodes, and that each sink node has $\text{maxflow}(t_k) = 2$, $(k = 1, \ldots, 6)$. The sink nodes receive data from different pair of intermediate nodes, u_i, $(i = 1, \ldots, 4)$. Our aim is all six sink nodes to be able to reconstruct the source packets that are exclusively coded in $GF(2)$.

Let us take the following Latin square and split it into two Latin rectangles:

$$L = \left[\begin{array}{cccc} 2 & 4 & 1 & 3 \\ 1 & 3 & 2 & 4 \\ 3 & 2 & 4 & 1 \\ \hline 4 & 1 & 3 & 2 \end{array}\right].$$

Each column from the 3×4 upper Latin rectangle represents a combination of source packets in a coded packet c_i, $i = 1, \ldots, 4$. Using the incidence matrix M of the Latin rectangle the source computes the coded packets.

Definition 4. *Let (X, A) be a design where $X = \{x_1, \ldots, x_v\}$ and $A = \{A_1, \ldots, A_b\}$. The incidence matrix of (X, A) is the $v \times b$ 0-1 matrix $M = (m_{i,j})$ defined by the rule* $m_{i,j} = \begin{cases} 1, & \text{if } x_i \in A_j, \\ 0, & \text{if } x_i \notin A_j. \end{cases}$

Proposition 1. *The incidence matrix $M = (m_{i,j})$ of any Latin rectangle with dimensions $k \times n$ is balanced matrix with k ones in each row and each column.*

Proof. From the definition of the incidence matrix it follows that the number of ones in each row is equal to the number of elements k in each column of the Latin rectangle. On the other hand, since each row of the Latin rectangle is a permutation of n elements, and there are no elements that occur twice in each column, the number of ones in each column can be neither less nor larger than k.

Note 1. The incidence matrix M of a $k \times n$ Latin rectangle is always balanced. However, the inverse matrix of the incidence matrix M^{-1} is not always balanced.

Proposition 2. *The necessary condition an incidence matrix $M = (m_{i,j})$ of a $k \times n$ Latin rectangle to be nonsingular in $GF(2)$ is k to be odd, i.e., $k = 2l+1$.*

Proof. Assume that k is even, i.e., $k = 2l$. Recall that a matrix M is nonsingular in $GF(2)$ if and only if its determinant is 1 (or it is singular if and only if its determinant is 0). Recall further the Leibniz formula for the determinant of an $n \times n$ matrix M: $det(M) = \sum_{\sigma \in S_n} sgn(\sigma) \prod_{i=1}^{n} m_{i,\sigma_i}$, where the sum is computed over all elements of the symmetric group of n elements S_n, i.e., over all permutations $\sigma \in S_n$, and $sgn(\sigma)$ is the signature (or the parity of the permutation) whose value is $+1$ or -1. The elements m_{i,σ_i} are the elements $m_{i,j}$ of the matrix M where the value for the index $j = \sigma_i$ is determined as the i–th element of the permutation σ.

If $k = 2l$ is even, from Proposition 1 and from the fact that operations are performed in $GF(2)$, it follows that every summand in the Leibniz formula gives an even number of nonzero products, thus the final sum must be even, i.e., the determinant in $GF(2)$ is 0.

The corresponding 4×4 incidence matrix of the Latin rectangle in Example 2 is nonsingular in $GF(2)$ (Proposition 2). M is represented as

$$M = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

A direct consequence from Theorem 1 is the following:

Corollary 1. *A sink node $t \in T$ with maxflow(t) can receive n source packets, encoded with the incidence matrix of a $k \times n$ Latin rectangle in $GF(2)$, in $p = \lceil \frac{n}{maxflow(t)} \rceil$ phases. In each phase the sink node reaches its maxflow(t).*

Following Corollary 1 the number of phases in which packets are sent depends from the total number of packets and maxflow(t_k).

Using M the source computes the vector of coded packets as

$$\mathbf{c} = M\mathbf{x} = [c_1, c_2, c_3, c_4]^\top$$

where $\mathbf{x} = [x_1, x_2, x_3, x_4]^\top$ is a vector of the source packets. The coded packets are XOR-ed combinations of the source packets, i.e.,

$$\begin{aligned} c_1 &= x_1 \oplus x_2 \oplus x_3, \\ c_2 &= x_2 \oplus x_3 \oplus x_4, \\ c_3 &= x_1 \oplus x_2 \oplus x_4, \\ c_4 &= x_1 \oplus x_3 \oplus x_4. \end{aligned}$$

The source further prepends the information from the incidence matrix to each of the coded packets. The vector of packets that are sent becomes as follows: $\mathbf{C} =$

Table 1. Description of receiving coded packets in each phase at the sink nodes

	t_1	t_2	t_3	t_4	t_5	t_6
First phase	C_4, C_1	C_4, C_3	C_4, C_3	C_1, C_2	C_1, C_2	C_3, C_2
Second phase	C_2, C_3	C_2, C_1	C_2, C_1	C_3, C_4	C_3, C_4	C_1, C_4

$\{(1,2,3,c_1),(2,3,4,c_2),(1,2,4,c_3),(1,3,4,c_4)\} = \{C_1, C_2, C_3, C_4\}$. The sink nodes receive in each phase a pair of different packets as shown in Table 1. Their buffer should be large enough to store the received packets C_i, $i = 1, \ldots, 4$. The decoding at the sink nodes is performed by M^{-1}. Each sink node computes M^{-1} from the prepended indexes. The original packets x_i, $i = 1, \ldots, 4$, are reconstructed as $\mathbf{x} = M^{-1}\mathbf{c}$. Note that although our approach is similar to [16], we use a systematic selection of the encoding functions and we do not send plain packets on the disjoint paths.

4 Additional Plausible Security Properties of the Balanced XOR-ed Coding

The work with incidence matrices related to randomly generated Latin rectangles is actually a work with balanced block designs. However, as we noted in Note 1, it is not necessary both the incidence matrix and its inverse matrix to be completely balanced. If we are interested in the complexity of decoding and the security issues when an adversary can successfully decode some sniffed packets, then the easiest way to address these issues is to give equal level of security to all encoded packets. In our approach this can be easily achieved by switching the roles of the incidence matrix and its inverse matrix: the encoding of the source packets is done with the inverse matrix of the incidence matrix and decoding of the coded packets is done with the incidence matrix. By applying this approach, decoding of any of the source packets requires an equal number of coded packets.

Corollary 2. *For each value of maxflow(t) and a number of source packets n which is multiple of maxflow(t), there exists a Latin rectangle with $n-1$ or $n-2$ rows and its incidence matrix can be used for decoding.*

Due to Proposition 2, when n is even the necessary requirement for a nonsingular incidence matrix is the Latin rectangle to have $n-1$ rows. When n is odd the necessary requirement for a nonsingular incidence matrix is the Latin rectangle to have $n-2$ rows.

Theorem 4. *When decoding is performed with the incidence matrix from Corollary 2, any eavesdropper needs to listen at least maxflow(t) links in order to decode at least one source packet.*

Proof. Assume that an adversary eavesdrops maxflow$(t) - 1$ links. Since the incidence matrix used for decoding is related to a Latin rectangle with $n-1$ or $n-2$ rows, eavesdropping "just" maxflow$(t) - 1$ links is not sufficient for the adversary to receive at least one subset of $n-1$ or $n-2$ packets from which he/she can decode at least one original packet.

Another remark that can be given about our approach is that the number of XOR operations between different packets (both in the source and in the sink nodes) is relatively high. We can address that remark by using Latin rectangles with smaller number of rows as a trade-off between the number of encoding/decoding operations and the ability of an adversary to decode a source packet. Namely, the encoding and decoding efforts at the source and sink node are the highest when encoding and decoding requires $n-1$ or $n-2$ packets. In order to decrease the number of operations at the nodes, the Latin rectangle should have $k \leq n-2$ rows. However, we are interested to reduce the number k without reducing the number of links that have to be listened by an eavesdropper in order to decode at least one original packet. The following theorem gives the necessary and sufficient condition for that to happen:

Theorem 5. *Let the coding be done by M^{-1} obtained from a Latin rectangle $L_{k\times n}$ of size $k \times n$, where $k \leq n-2$. Further, assume that the transfer is done by sending n packets from s to t in $p = \lceil \frac{n}{maxflow(t)} \rceil$ phases on $maxflow(t)$ disjoint paths and let the sets of indexes of the packets sent via i-th disjoint path are denoted by $S_i, i = 1, \ldots, maxflow(t)$. A necessary and sufficient condition for an eavesdropper to need to listen at least $maxflow(t)$ links in order to decode at least one original packet is:*

$$\forall j \in \{1, \ldots, n\}, \forall i \in \{1, \ldots, maxflow(t)\} : L_{k,j} \cap S_i \neq \emptyset, \tag{1}$$

where $L_{k,j}, j \in \{1, \ldots, n\}$ is the set of elements in the j-th column of the Latin rectangle $L_{k\times n}$.

Proof. To show that the condition (1) is necessary assume that an eavesdropper needs $\text{maxflow}(t) - 1$ links in order to decode one original packet x_l, and let us denote by S_m the set of indexes of the packets sent via the disjoint path that was not listened by the eavesdropper. This means that for the l-th column $L_{k,l}$ of the Latin rectangle $L_{k\times n}$: $L_{k,l} \cap S_m = \emptyset$ which violates the condition (1).

To show that the condition (1) is sufficient, let us denote by $S_{i,j} = L_{k,j} \cap S_i$, $j \in \{1, \ldots, n\}, i \in \{1, \ldots, \text{maxflow}(t)\}$. It is sufficient to notice that $S_{i,j}$ are disjunctive partitions for every set $L_{k,j}$, i.e.,

$$\forall j \in \{1, \ldots, n\} : \bigcup_{i=1}^{\text{maxflow}(t)} S_{i,j} = L_{k,j}$$

and

$$\forall j_1, j_2 \in \{1, \ldots, n\} : S_{i,j_1} \cap S_{i,j_2} = \emptyset.$$

Since $|L_{k,j}| = k$, and the encoding of original n packets is done by M^{-1}, it follows that an eavesdropper can decode any original packet only by listening at least $\text{maxflow}(t)$ links.

Example 3. We present an example that illustrates the security in our approach. The goal is to achieve secrecy[1] so that a passive adversary is able to reconstruct n source packets only when at least maxflow(t) links are eavesdropped. By sending XOR-ed packets on disjoint paths (exploiting the path diversity), an adversary is unable to decode the message although several paths are eavesdropped. Let us consider the network shown in Fig.3, where a source s communicates with two sinks t_1 and t_2 with the help of intermediate nodes u_i, $i = 1, 2, 3$, and sends twelve packets to t_1 and t_2. Packets are sent in four phases since maxflow(t) = 3. Let us use the following 5×12 Latin rectangle:

$$L_{5\times 12} = \begin{bmatrix} 4 & 2 & 11 & 8 & 12 & 1 & 9 & 5 & 10 & 7 & 6 & 3 \\ 2 & 8 & 12 & 3 & 6 & 10 & 4 & 11 & 5 & 1 & 9 & 7 \\ 3 & 4 & 2 & 9 & 11 & 12 & 5 & 6 & 7 & 8 & 10 & 1 \\ 9 & 10 & 1 & 6 & 3 & 7 & 2 & 8 & 4 & 11 & 12 & 5 \\ 6 & 5 & 7 & 10 & 1 & 11 & 8 & 3 & 12 & 4 & 2 & 9 \end{bmatrix}.$$

The colors of indexes in $L_{5\times 12}$ correspond to the colors of the packets as they are sent in Fig 3. If the sink nodes reconstruct the source packets with M^{-1}, then not all packets have the same level of decoding complexity. That is demonstrated with relations (2) and (3). For instance, to decode x_4, x_7 and x_8 nine coded packets are needed, while to decode x_5 and x_{11} just three packets are needed. The goal is to avoid this non-balanced complexity in the decoding. Therefore, as in Theorem 5 the encoding is done by M^{-1} and the decoding by M. When s computes the vector of coded packets as $\mathbf{c} = M^{-1}\mathbf{x}$, then the coded packets c_i, $i = 1, \ldots, 12$ are XOR-ed combinations of different number of source packets. Consequently, decoding of packets is done with a balanced matrix, i.e., $\mathbf{x} = M\mathbf{c}$.

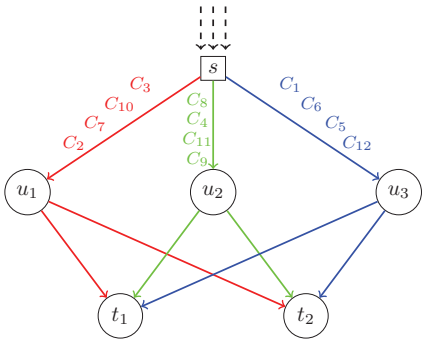

Fig. 3. Routing of 12 packets for secure coding when decoding is performed with 5 coded packets

[1] We use here the term *secrecy* as it is used in [13, Ch.7 pp. 185]

$$
\begin{aligned}
c_1 &= x_2 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_9,\\
c_2 &= x_2 \oplus x_4 \oplus x_5 \oplus x_8 \oplus x_{10},\\
c_3 &= x_1 \oplus x_2 \oplus x_7 \oplus x_{11} \oplus x_{12},\\
c_4 &= x_3 \oplus x_6 \oplus x_8 \oplus x_9 \oplus x_{10},\\
c_5 &= x_1 \oplus x_3 \oplus x_6 \oplus x_{11} \oplus x_{12},\\
c_6 &= x_1 \oplus x_7 \oplus x_{10} \oplus x_{11} \oplus x_{12},\\
c_7 &= x_2 \oplus x_4 \oplus x_5 \oplus x_8 \oplus x_9,\\
c_8 &= x_3 \oplus x_5 \oplus x_6 \oplus x_8 \oplus x_{11},\\
c_9 &= x_4 \oplus x_5 \oplus x_7 \oplus x_{10} \oplus x_{12},\\
c_{10} &= x_1 \oplus x_4 \oplus x_7 \oplus x_8 \oplus x_{11},\\
c_{11} &= x_2 \oplus x_6 \oplus x_9 \oplus x_{10} \oplus x_{12},\\
c_{12} &= x_1 \oplus x_3 \oplus x_5 \oplus x_7 \oplus x_9.
\end{aligned}
\tag{2}
$$

$$
\begin{aligned}
x_1 &= c_2 \oplus c_5 \oplus c_{10} \oplus c_{11} \oplus c_{12},\\
x_2 &= c_1 \oplus c_3 \oplus c_5 \oplus c_6 \oplus c_7 \oplus c_9 \oplus c_{10},\\
x_3 &= c_3 \oplus c_5 \oplus c_6 \oplus c_7 \oplus c_8 \oplus c_9 \oplus c_{12},\\
x_4 &= c_4 \oplus c_5 \oplus c_6 \oplus c_7 \oplus c_8 \oplus c_9 \oplus c_{10} \oplus c_{11} \oplus c_{12},\\
x_5 &= c_1 \oplus c_2 \oplus c_4,\\
x_6 &= c_1 \oplus c_2 \oplus c_4 \oplus c_6 \oplus c_7 \oplus c_{10} \oplus c_{11},\\
x_7 &= c_2 \oplus c_3 \oplus c_4 \oplus c_5 \oplus c_6 \oplus c_7 \oplus c_8 \oplus c_{11} \oplus c_{12},\\
x_8 &= c_1 \oplus c_3 \oplus c_5 \oplus c_7 \oplus c_8 \oplus c_9 \oplus c_{10} \oplus c_{11} \oplus c_{12},\\
x_9 &= c_1 \oplus c_2 \oplus c_5 \oplus c_9 \oplus c_{10},\\
x_{10} &= c_1 \oplus c_5 \oplus c_7 \oplus c_9 \oplus c_{10},\\
x_{11} &= c_1 \oplus c_7 \oplus c_8,\\
x_{12} &= c_3 \oplus c_4 \oplus c_5 \oplus c_7 \oplus c_9.
\end{aligned}
\tag{3}
$$

Assume that the routing is as follows: on the first path the source sends (C_3, C_{10}, C_7, C_2), on the second path (C_8, C_4, C_{11}, C_9) and (C_1, C_6, C_5, C_{12}) on the third path as it is shown in Fig.3. We use three different colors for the packets sent to three disjoint paths in order to demonstrate the essence of the proof of Theorem 5. Note that all colors are present in every column of the Latin rectangle $L_{5\times 12}$. This corresponds to the condition (1) in Theorem 5. In order to reconstruct at least one source packet, an adversary must eavesdrop at least 3 links.

5 Conclusions

In this paper we have presented a construction of codes over $GF(2)$ which reach the max-flow for single source multicast acyclic networks with delay. The coding is exclusively performed in $GF(2)$, i.e., it is a bitwise XOR of packets with equal lengths. The encoding and decoding are based on balanced nonsingular matrices that are obtained as incidence matrices from Latin rectangles. Balanced XOR-ed coding is of particular importance for energy and processor constraint devices. Additionally, we showed that the approach offers plausible security properties, i.e., if an eavesdropper wants to reconstruct at least one original packet, then the number of eavesdropped links must be equal to the max-flow of the network.

Possible future work includes intermediate nodes to form coded packets, as well as building networks dynamically by adding more and more sink nodes that reach the max-flow when the coding is XOR-ed coding.

Acknowledgements

We would like to thank Gergely Biczók for his discussions and remarks that significantly improved the paper.

References

1. Ahlswede, R., Cai, N., Li, S.Y.R., Yeung, R.W.: Network information flow. IEEE Transactions on Information Theory. vol. 46, pp. 1204–1216 (2000)

2. Bhattad, K., Narayanan, K.R.: Weakly secure network coding. In: Proc. First Workshop on Network Coding, Theory, and Applications (NetCod) (2005)
3. Colbourn, C.J., Dinitz, J.H.: Handbook of Combinatorial Designs. Chapman, Hall/CRC (2006)
4. Colbourn, C.J., Dinitz, J. H., Stinson, D.R.: Applications of combinatorial designs to communications, cryptography, and networking (1999)
5. Hall, P.: On representatives of subsets. J. London Math. Soc. 37, 26–30 (1935)
6. Ho, T., Médard, M., Koetter, R., Karger, D.R., Effros, M., Shi, J., Leong, B.: A random linear network coding approach to multicast. IEEE Transactions on Information Theory. vol.52, pp. 4413–4430 (2006)
7. Jacobson, M.T., Matthews, P.: Generating uniformly distributed random latin squares. Journal of Combinatorial Designs 6, 405–437 (1996)
8. Jaggi, S., Cassuto, Y., Effros, M.: Low complexity encoding for network codes. In: IEEE International Symposium on Information Theory. pp. 40–44 (2006)
9. Katti, S., Rahul, H., Hu, W., Katabi, D., Médard, M., Crowcroft, J.: XORs in the air: Practical wireless network coding. IEEE/ACM Trans. Netw. 3, 497–510 (2008)
10. Koetter, R., Médard, M.: An algebraic approach to network coding. IEEE/ACM Trans. Netw. 5, 782–795 (2003)
11. Lawler, E.: Combinatorial Optimization: Networks and Matroids. Dover Publications (2001)
12. Li, S.Y.R., Yeung, R.W., Cai, N.: Linear network coding. IEEE Transactions on Information Theory 2, 371–381 (2003)
13. Médard, M., Sprintson, A.: Network coding, Fundamentals and Applications. Academic Press (2012)
14. Pedersen, M.V., Fitzek, F.H.P., Larsen, T.: Implementation and performance evaluation of network coding for cooperative mobile devices. In: IEEE Cognitive and Cooperative Wireless Networks Workshop (2008)
15. Qureshi, J., Foh, C.H., Cai, J.: Optimal solution for the index coding problem using network coding over gf(2). In: IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON) 209–217 (2012)
16. Riis, S.: Linear versus nonlinear boolean functions in network flow. CISS (2004)
17. Rouayheb, S.Y.E., Sprintson, A., Georghiades, C.N.: On the index coding problem and its relation to network coding and matroid theory. submitted to IEEE Transactions on Information Theory (2008)
18. Shojania, H., Li, B.: Random network coding on the iphone: fact or fiction?. NOSSDAV (2009)
19. Stinson, D. R.: Combinatorial Designs: Constructions and Analysis. SpringerVerlag (2003)
20. Vingelmann, P., Pedersen, M.V., Fitzek, F.H.P., Heide, J.: Multimedia distribution using network coding on the iphone platform. In: Proceedings of the 2010 ACM multimedia workshop on Mobile cloud media computing (2010)