Shortening Array Codes and the Perfect 1-Factorization Conjecture

Vasken Bohossian Jehoshua Bruck

California Institute of Technology
Mail Code 136-93
Pasadena, CA 91125
E-mail: {vincent, bruck }@paradise.caltech.edu

***Abstract*— The existence of a perfect 1-factorization of the complete graph K_n, for arbitrary n, is a 40-year old open problem in graph theory. Two infinite families of perfect 1-factorizations are known for K_{2p} and K_{p+1}, where p is a prime. It was shown in [8] that finding a perfect 1-factorization of K_n can be reduced to a problem in coding, i.e. to constructing an MDS, lowest density array code of length n. In this paper, a new method for shortening arbitrary array codes is introduced. It is then used to derive the K_{p+1} family of perfect 1-factorizations from the K_{2p} family, by applying the reduction metioned above. Namely, techniques from coding theory are used to prove a new result in graph theory.**

I. Introduction

Array Codes are erasure-correcting codes represented by an array of bits. Erasures correspond to the loss of columns. A two-erasure correcting array code, for example, is capable of recovering any two lost columns. For a survey on array codes see [4]. For recent results in array codes see [1], [2], [3], [6].

Example 1 (Simple Array Code): A simple two-erasure correcting array code of length four is shown below:

a	b	c	d
$b+c$	$c+d$	$d+a$	$a+b$

The first row consists of four information bits a, b, c and d. The second row contains four parity bits. The '+' sign indicates bitwise exclusive-OR, so that $x + x = 0$. One can verify that any two columns can recover all four information bits. Suppose, for example, that columns three and four are lost:

a	b		
$b+c$	$c+d$		

c can be recovered by adding $b + c$ to b:

$$c = (b + c) + b$$

d can be recovered using $c + d$ and c:

$$d = (c + d) + c$$

Similar decoding chains are used for other erasure patterns.

The B-Code, introduced in [10] (only the K_{p+1} infinite family of codes), and in [8] (both K_{p+1} and K_{2p} families), is a two-erasure correcting array code of length $2n$, represented by a n by $2n$ array. It can recover any two out of $2n$ lost columns. The construction of the B-Code is based on the perfect 1-factorization of the complete graph, K_{2n}.

Definition 1 (Perfect 1-factorization): A perfect 1-factorization of a graph is a partitioning of the set of its edges into subsets, called factors, such that each factors is a graph of degree one, and the union of any two factors forms a Hamiltonian cycle.

In the case of the complete graph of even size, K_{2n}, it is still unknown whether or not a perfect 1-factorization exists for all values of n ([5], [7]). The following was conjectured in 1963:

Conjecture 1 (Perfect 1-factorization): A perfect 1-factoriztion of the complete graph K_{2n} exists for all values of n, $n > 1$.

So far, two infinite families of perfect 1-factorizations have been shown to exists, namely, the factorizations of K_{p+1} and K_{2p}, where p is an arbitrary prime number ($p > 2$).

The contributions of this paper are twofold:

- A method for **shortening** the B-Code is introduced. It could be used in general to shorten an arbitrary array code.
- The method along with additional manipulation (**separation**) is used to derive the perfect 1-factorization of the complete graph K_{p+1} from the perfect 1-factorization of K_{2p}. The derivation consists of the following steps:

1) $\mathcal{P}_{2p}$: perfect 1-factorization of K_{2p}, obtained by known construction ([5], [7]). Shown in Section II-A.
2) $\mathcal{P}_{2p} \Longrightarrow B_{2p-1}$: extended B-Code of length $2p - 1$, by known construction from [8]. Section II-B.
3) $B_{2p-1} \Longrightarrow \tilde{X}_p$: generalized X-Code of length p, by **shortening**: new construction. Section IV-A.
4) $\tilde{X}_p \Longrightarrow B_p$: extended B-Code of length p, by **separation**: new construction. Section IV-B.
5) $B_p \Longrightarrow \mathcal{P}_{p+1}$, by Theorem 5 in [8].

The above steps are illustrated, in Section III, by examples for $p = 5$. Proofs are provided for arbitrary p, for steps 3 and 4 (Sections IV-A and IV-B).

II. Constructions

In this section we summarize two known constructions needed for sections III and IV, namely the perfect 1-factorization of K_{2p}, from [7] and the B-Code construction from [8].

A. $\mathcal{P}_{2p}$: Perfect 1-factorization of K_{2p}

For any prime number p the complete graph of $2p$ vertices, K_{2p}, has a perfect 1-factorization, $\mathcal{P}_{2p}$.

Construction 1 (General case: $\mathcal{P}_{2p}$): The construction of $\mathcal{P}_{2p}$ is as follows: there are a total of $2p-1$ factors, f_i, of which one is the principal factor, f_p, $p-1$ are even numbered factors and $p-1$ are odd numbered factors. Below is the formal definition of the construction. Note that a factor is labeled f_i if it contains edge $(0,i)$.

$$\mathcal{P}_{2p} = \{f_i\}, \textit{ for } i \in \{1,..,2p-1\}, \textit{ where:}$$

$$f_i = \begin{cases} \{(0,p), e_{0,1}, e_{0,2}, .., e_{0,p-1}\} & \textit{, for } i = p \\ \{(\frac{i}{2}, \frac{i}{2}+p), e_{i,1}, e_{i,2}, .., e_{i,p-1}\} & \textit{, for } i \textit{ even, } i \neq 0 \\ \{e_{i,0}, e_{i,1}, e_{i,2}, .., e_{i,p-1}\} & \textit{, for } i \textit{ odd, } i \neq p \end{cases}$$

$$e_{i,j} = \begin{cases} ((\frac{i}{2}-j) \bmod 2p, (\frac{i}{2}+j) \bmod 2p) & \textit{, for } i \textit{ even} \\ (2j, (2j+i) \bmod 2p) & \textit{, for } i \textit{ odd} \end{cases}$$

Where $e_{i,j}$ is the edge between vertices i and j. The following tables show $\mathcal{P}_{2p}$ as a list of edges per factor. All entries are modulo $2p$. Principal factor f_p:

f_p
$0,p$
$-1,1$
$-2,2$
$\vdots$
$-p+2,p-2$
$-p+1,p-1$

Even factors f_i, i even:

f_2	f_4	...	f_{2p-2}
$1,1+p$	$2,2+p$	...	$p-1,-1$
$0,2$	$1,3$	...	$p-2,0$
$-1,3$	$0,4$	...	$p-3,1$
$\vdots$	$\vdots$	...	$\vdots$
$-p+3,p-1$	$-p+4,p$	...	$1,-3$
$-p+2,p$	$-p+3,p+1$	...	$0,-2$

Odd factors f_i, i odd, $i \neq p$:

f_1	f_3	...	f_{p-2}	f_{p+2}	...	f_{2p-1}
$0,1$	$0,3$	...	$0,p-2$	$0,p+2$	...	$0,-1$
$2,3$	$2,5$	...	$2,p$	$2,p+4$	...	$2,1$
$4,5$	$4,7$	...	$4,p+2$	$4,p+6$	...	$4,3$
$\vdots$	$\vdots$	...	$\vdots$	$\vdots$	...	$\vdots$
$-4,-3$	$-4,-1$	...	$-4,p-6$	$-4,p-2$	...	$-4,-5$
$-2,-1$	$-2,1$	...	$-2,p-4$	$-2,p$	...	$-2,-3$

Notice that the odd factors have edges of odd length, while the principal factor and the even factors have even-length edges, with the exception of exactly one edge per factor, the first one, which is of length p.
For the proof that $\mathcal{P}_{2p}$ is indeed a perfect 1-factorization see [7].

B. Erasure-Correcting Code based on $\mathcal{P}_{2p}$

To each perfect 1-factorization of size $2p$, correspond two erasure-correcting array codes: the B-Code, of size $2p-2$, and the extended B-Code of size $2p-1$, B_{2p-2} and B_{2p-1} respectively [8].

Construction 2 (From P_{2p} to B_{2p-1}): The construction of B_{2p-1} is as follows: there are a total of $2p-1$ columns, of which one is a column of pure information bits corresponding to the edges of the principal factor, f_p, of $\mathcal{P}_{2p}$. The remaining $2p-2$ columns correspond to $p-1$ even factors and $p-1$ odd factors. They each contain one parity bit, corresponding to vertex i, and $p-2$ information bits, corresponding to the other edges in f_i. The parity bit is the sum of all information bits corresponding to edges connected to vertex i. Here is the formal definition of the construction:

$$B_{2p-1} = \{b_i\}, \textit{ for } i \in \{1,..,2p-1\}, \textit{ where:}$$

$$b_i = \begin{cases} \{\hat{e}_{0,1}, \hat{e}_{0,2}, .., \hat{e}_{0,p-1}\} & \textit{, for } i = p \\ \{\hat{p}_i, (\frac{i}{2}, \frac{i}{2}+p), \hat{e}_{i,1}, \hat{e}_{i,2}, .., \hat{e}_{i,p-1}\} & \textit{, for } i \textit{ even, } i \neq 0 \\ \{\hat{p}_i, \hat{e}_{i,0}, \hat{e}_{i,1}, \hat{e}_{i,2}, .., \hat{e}_{i,p-1}\} & \textit{, for } i \textit{ odd, } i \neq p \end{cases}$$

$$\hat{p}_i = \sum_{(j,k)/i \in e_{j,k}} \hat{e}_{j,k}$$

$$\hat{e}_{i,j} = \begin{cases} \emptyset & \textit{, if } 0 \in e_{i,j} \\ \emptyset & \textit{, if } p \in e_{i,j} \\ I_{e_{i,j}} & \textit{, otherwise} \end{cases}$$

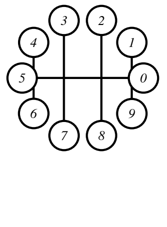
$$e_{i,j} = \begin{cases} \{(\frac{i}{2}-j) \bmod 2p, (\frac{i}{2}+j) \bmod 2p\} & \textit{, for } i \textit{ even} \\ \{2j, (2j+i) \bmod 2p\} & \textit{, for } i \textit{ odd} \end{cases}$$

Where $e_{i,j}$ are the edges of $\mathcal{P}_{2p}$. $I_{e_{i,j}}$ (and $\hat{e}_{i,j}$) are the information bits. They correspond to the edges $e_{i,j}$ minus edges connected to vertices 0 and p. $\hat{p}_i$ is the parity bit in column i. It is defined as the sum of all information bits $\hat{e}_{j,k}$, such that $i \in e_{j,k}$. Notice that by definition every information bit appears in exactly two parity bits.

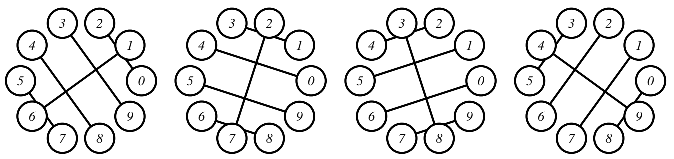
III. Examples

Example 2 ($\mathcal{P}_{10}$: perfect 1-factorization of K_{10}): The factorization consists of nine factors with five edges per factor. It is shown below in both graph and table formats. The factors are indexed by the number of the vertex connected to vertex 0.

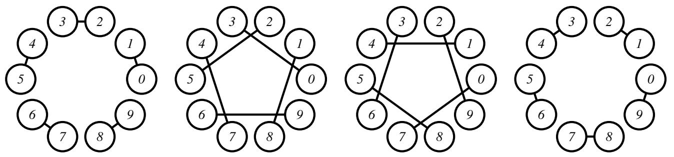
Principal factor, f_5:



Even factors f_2, f_4, f_6, f_8:



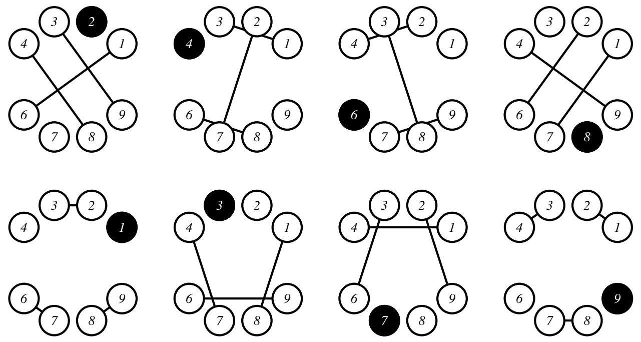
Odd factors f_1, f_3, f_7, f_9:



f_5	f_2	f_4	f_6	f_8	f_1	f_3	f_7	f_9
$0,5$	$1,6$	$2,7$	$3,8$	$4,9$	$0,1$	$0,3$	$0,7$	$0,9$
$9,1$	$0,2$	$1,3$	$2,4$	$3,5$	$2,3$	$2,5$	$2,9$	$2,1$
$8,2$	$9,3$	$0,4$	$1,5$	$2,6$	$4,5$	$4,7$	$4,1$	$4,3$
$7,3$	$8,4$	$9,5$	$0,6$	$1,7$	$6,7$	$6,9$	$6,3$	$6,5$
$6,4$	$7,5$	$8,6$	$9,7$	$0,8$	$8,9$	$8,1$	$8,5$	$8,7$

Reminder: factors are labeled f_i, where i is the vertex connected to 0.

Example 3 (B_8: B-Code of size 8): We construct B_8 from $\mathcal{P}10$ by deleting the principal factor f_5, as well as vertices 0 and 5 and all edges connected to them. Here are the resulting factors in graph format:



Vertices correspond to parity bits, while edges are information bits. Black verices are the ones that were connected to vertex 0. They indicate the placement of parity bits relative to columns of information bits. Notice that the union of any two factors forms a graph such that starting at the black nodes and following edges one can uniquely traverse all remaining nodes (this is the erasure-recovery path). The array representation of the above graphs is obtained from the table of Example 2:

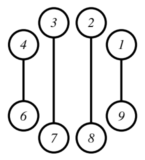
$a_{1,6}$	$a_{2,7}$	$a_{3,8}$	$a_{4,9}$	$a_{2,3}$	$a_{4,7}$	$a_{2,9}$	$a_{2,1}$
$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$	$a_{6,7}$	$a_{6,9}$	$a_{4,1}$	$a_{4,3}$
$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$	$a_{8,9}$	$a_{8,1}$	$a_{6,3}$	$a_{8,7}$
p_2	p_4	p_6	p_8	p_1	p_3	p_7	p_9

The $a_{j,k}$ are information bits. The p_i are parity bits. They are related by:

$$p_i = \sum_{(j,k)/i\in\{j,k\}} a_{j,k}$$

In other words, parity bit i is the sum of all information bits that have i as one of their two indices. Notice that by definition every information bit appears in exactly two parity bits.

Example 4 (Extended B-Code of size 9, B_9): B_8 can be extended by the addition of a column of information bits. Those are the bits corresponding to the edges of the principal factor f_5 (which was deleted in the construction of B_8):



$a_{1,9}$	$a_{1,6}$	$a_{2,7}$	$a_{3,8}$	$a_{4,9}$	$a_{2,3}$	$a_{4,7}$	$a_{2,9}$	$a_{2,1}$
$a_{2,8}$	$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$	$a_{6,7}$	$a_{6,9}$	$a_{4,1}$	$a_{4,3}$
$a_{3,7}$	$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$	$a_{8,9}$	$a_{8,1}$	$a_{6,3}$	$a_{8,7}$
$a_{4,6}$	p_2	p_4	p_6	p_8	p_1	p_3	p_7	p_9

Example 5 (Shortening B_8 into $\tilde{X}_4$): In the array representing B_8 (from Example 3) we set all information bits in the last 4 columns to zero. We obtain the following array:

$a_{1,6}$	$a_{2,7}$	$a_{3,8}$	$a_{4,9}$	0	0	0	0
$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$	0	0	0	0
$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$	0	0	0	0
p_2	p_4	p_6	p_8	p_1	p_3	p_7	p_9

Notice that the zeroed columns correspond to parities with odd indices. By definition those parities can now be written as:

$$\begin{array}{ccccccc} p_1 & = & a_{1,6} & + & a_{1,3} & + & a_{1,7} \\ p_3 & = & a_{9,3} & + & a_{1,3} & + & a_{3,8} \\ p_7 & = & a_{2,7} & + & a_{9,7} & + & a_{1,7} \\ p_9 & = & a_{9,3} & + & a_{9,7} & + & a_{4,9} \end{array}$$

Notice that each equation has exactly one information bit, $a_{i,j}$, with an even index. Rewriting the above equations we get:

$$\begin{array}{ccccccc} a_{1,6} & = & p_1 & + & a_{1,3} & + & a_{1,7} \\ a_{3,8} & = & a_{9,3} & + & a_{1,3} & + & p_3 \\ a_{2,7} & = & p_7 & + & a_{9,7} & + & a_{1,7} \\ a_{4,9} & = & a_{9,3} & + & a_{9,7} & + & p_9 \end{array}$$

Renaming $a_{1,6}$, $a_{3,8}$, $a_{2,7}$ and $a_{4,9}$ as parities and p_1, p_3, p_7 and p_9 as information bits, we set the information bits to zero and relabel:

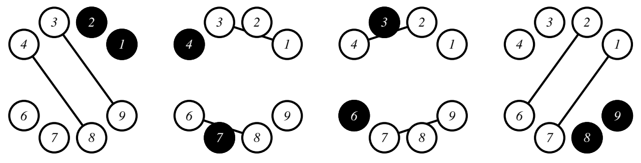
p_1	p_7	p_3	p_9	0	0	0	0
$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$	0	0	0	0
$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$	0	0	0	0
p_2	p_4	p_6	p_8	0	0	0	0

Notice that, because of the change of variables, the even-numbered parities depend on two extra information bits. This fact will be ignored in the graphical representation, but will be taken into account in the final proof.

Rearranging and removing the zeroed columns we get the array representing $\tilde{X}_4$:

$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$
$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$
p_1	p_7	p_3	p_9
p_2	p_4	p_6	p_8

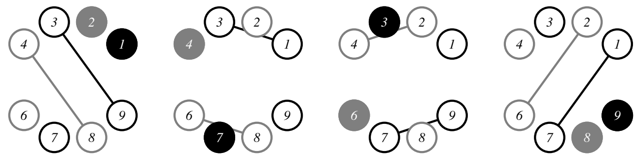
In the graph domain each factor has two edges and two black vertices:



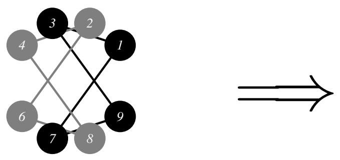
Example 6 (Shortening B_9 into $\tilde{X}_5$): B_9 has an extra column of information bits. Those bits are of the form $a_{i,j}$ where i and j are either both odd, or both even. Therefore they do not interfere with the choice of information bits that are substituted with parities. The resulting array for $\tilde{X}_9$ is:

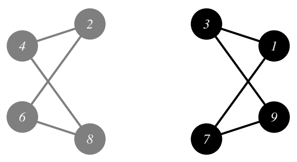
$a_{1,9}$	$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$
$a_{2,8}$	$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$
$a_{3,7}$	p_1	p_7	p_3	p_9
$a_{4,6}$	p_2	p_4	p_6	p_8

Example 7 (Separation of $\tilde{X}_4$): Notice that the arrays for $\tilde{X}_4$ and $\tilde{X}_5$ above contain edges of only even length. We color grey all even nodes and edges touching them. All odd nodes and edges – black:



Putting everything toghether and disentangling:





In array from the separation looks like:

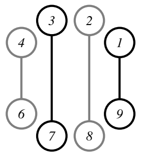
$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$
$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$
p_1	p_7	p_3	p_9
p_2	p_4	p_6	p_8

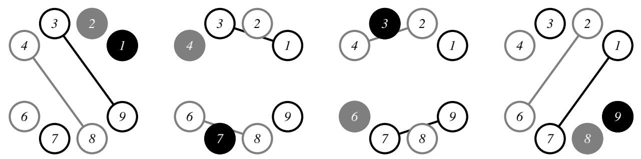
$\Longrightarrow$

$a_{9,3}$	$a_{1,3}$	$a_{9,7}$	$a_{1,7}$
p_1	p_7	p_3	p_9

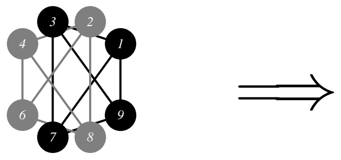
$a_{8,4}$	$a_{8,6}$	$a_{2,4}$	$a_{2,6}$
p_2	p_4	p_6	p_8

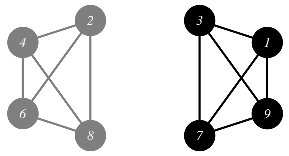
Example 8 (Separation of $\tilde{X}_5$): Here is the process with the extra information column:





Putting everything toghether and disentangling:





In array from the separation looks like:

$a_{1,9}$	$a_{9,3}$	$a_{1,3}$	$a_{2,4}$	$a_{2,6}$
$a_{2,8}$	$a_{8,4}$	$a_{8,6}$	$a_{9,7}$	$a_{1,7}$
$a_{3,7}$	p_1	p_7	p_3	p_9
$a_{4,6}$	p_2	p_4	p_6	p_8

$\Longrightarrow$

$a_{1,9}$	$a_{9,3}$	$a_{1,3}$	$a_{9,7}$	$a_{1,7}$
$a_{3,7}$	p_1	p_7	p_3	p_9

$a_{2,8}$	$a_{8,4}$	$a_{8,6}$	$a_{2,4}$	$a_{2,6}$
$a_{4,6}$	p_2	p_4	p_6	p_8

IV. Theorems and Proofs

A. Shortening: $B_{2p-1} \Longrightarrow \tilde{X}_p$

Let $n = p - 1$. B_{2n} is represented by a $n \times 2n$ array. By setting n^2 information bits to zero, the array can be shortened into a $n \times n$ array corresponding to a new, square-shaped array code of size n. It has the dimmensions of the X-Code [9]. We call this new code "generalized" X-Code of size n and denote it by $\tilde{X}_n$.

Construction 3 (Shortening B_{2p-2} and B_{2p-1}): Referring to Construction 2:

1) *In the columns corresponding to odd factors set all information bits to zero:*

$$I_{e_{i,j}} = 0, \text{ for } i \text{ odd}, \ i \neq p$$

2) *For the parity bit in each zeroed column identify an information bit in a non-zeroed column and exchange them by a change of variables:*

$$\hat{p}_i = \hat{e}_{j,k} + S_i \implies \hat{e}_{j,k} = \hat{p}_i + S_i$$

where $\hat{e}_{j,k}$ is interpreted as a new parity bit and $\hat{p}_i$ a new information bit. S_i is the sum of the remaining information bits in the original $\hat{p}_i$.

3) *Set the new information bits to zero:*

$$\hat{p}_i = 0$$

Theorem 1 ($\tilde{X}_{p-1}$ and $\tilde{X}_p$ are MDS): The shortened B-Codes, $\tilde{X}_{p-1}$ and $\tilde{X}_p$, obtained by the construction described above are MDS.

Proof: The proof consists of two parts. We first show that a single change of variables between a parity bit and an information bit preserves the MDS properties of the array. We then show that to the parity bit, $\hat{p}_i$, in every zeroed column uniquely corresponds one non-zeroed column, and an information bit, $\hat{e}_{j,k}$, in it, such that:

$$\hat{p}_i = \hat{e}_{j,k} + S_i$$

Part 1: A single change of variables descibed in the construction above corresponds to adding a row to another row of the parity check matrix of the B-Code, thus preserving the MDS property of the code.

Part 2: Consider the information bit indexed by:

$$e = \{\frac{i}{2}, \frac{i}{2} + p\}$$

(see Construction 2). One such bit appears in each even numbered column (i even) of the array. p being and odd prime implies that either $\frac{i}{2}$ is odd or $\frac{i}{2} + p$ is odd, but not both. Therefore I_e the bit indexed by edge e, appears in exactly one parity bit p_k, k odd, i.e. in exactly one of the zeroed columns.

B. Separation: from $\tilde{X}_p$ to B_p

Because of the particular shortening used in Section IV-A, we show that each column, C_i, of $\tilde{X}_p$ divides into two sets of bits A_i and B_i, such that the parity bit of set A_i only depends on information bits in sets A_j (and not on information bits of any of the B_j sets). We can therefore extract a new array based on the A_i, which turns out to be the array representing the B-Code, B_p. Here follows the formal theorem and proof:

Theorem 2 (Separation of $\tilde{X}_p$): $\tilde{X}_p]$ can be separated into two arrays, of which one corresponds to B_p.

Proof: By examining Construction 2, notice that after shortening the B-Code, B_{2p-1}, we are left only with edges of even length. In other words, all information bits in $\tilde{X}_p$ are indexed by pairs of the form:

$$e = \{(\frac{i}{2} - j) \bmod 2p, (\frac{i}{2} + j) \bmod 2p\}$$

Therefore, half of the information bits of every column in $\tilde{X}_p$ are of the form $a_{i,j}$ where both i and j are odd. For the other half both i and j are even. By definition, the even-indexed parities in B_{2p-1} depend only on information bits with at least one even index. That is true also for the new parities of $\tilde{X}_p$, defined by the change of variable during the shortening process. Therefore, all even-indexed bits form an independent $p \times \frac{p-1}{2}$ array such that every information bit appears in exactly two parity bits. As mentioned in Example 5, the even parities depend on some odd information bits. Those can be set to zero. The resulting code is MDS because $\tilde{X}_p$ is MDS. A counting argument shows that such a code can only be B_p. Indeed the number of parity bits (nodes) is $p-1$ and the number of information bits (edges) is: $p\frac{p-1}{2} - (p-1) = \frac{(p-1)(p-2)}{2}$ That is the number of edges in the complete graph K_{p-1}.

V. Conclusion

We presented a method for shortening the B-Code and used it to derive the perfect 1-factorization of the complete graph K_{p+1} from the perfect 1-factorization of K_{2p}. The construction consists of the following steps:

1) $\mathcal{P}_{2p}$: perfect 1-factorization of K_{2p}, obtained by known construction ([5], [7]). Shown in Section II-A.
2) $\mathcal{P}_{2p} \Longrightarrow B_{2p-1}$: extended B-Code of length $2p-1$, by known construction from [8]. Section II-B.
3) $B_{2p-1} \Longrightarrow \tilde{X}_p$: generalized X-Code of length p, by **shortening**: new construction. Section IV-A.
4) $\tilde{X}_p \Longrightarrow B_p$: extended B-Code of length p, by **separation**: new construction. Section IV-B.
5) $B_p \Longrightarrow \mathcal{P}_{p+1}$, by Theorem 5 in [8].

References

[1] M. Blaum, J. Brady, J. Bruck, and J. Menon, "EVENODD: an efficient scheme for tolerating multiple disk failures in RAID architectures", *IEEE Transactions on Computers*, 44(2):192-202, 1995.

[2] M. Blaum, J. Bruck, and A. Vardy, "MDS array codes with independent parity symbols", *IEEE-Trans-IT*, 42(2):529-542, 1996.

[3] M. Blaum and R. M. Roth, "On lowest density MDS codes", *IEEE-Trans-IT*, 45(1):46-59, 1999.

[4] P. G. Farrell, "A Survey of Array Error Control Codes", *ETT* , Volume 3, Number 5, pages 441-454, 1992.

[5] A. Kotzig, "Hamilton Graphs and Hamilton Circuits", *Theory of Graphs and Its Applications*, Proc. Sympos. Smolenice, pages 63-82, 1963.

[6] E. Luidor and R. M. Roth, "Lowest-density MDS codes over extension alphabets", *Technion CS Technical report*, available at http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2005/CS/CS-2005-09.pdf

[7] W. D. Wallis, *One-Factorizations*, Kluwer Academic Publisher, 1997.

[8] L. Xu, V. Bohossian, J. Bruck, and D. Wagner, "Low Density MDS Codes and Factors of Complete Graphs", *IEEE-Trans-IT*, Volume 45, Number 6, page 1817-1826, September 1999.

[9] L. Xu and J. Bruck, "X-Code: MDS Array Codes with Optimal Encoding" *IEEE-Trans-IT*, Volume 45, Number 1, pages 272-276, January 1999.

[10] G. V. Zaitsev, V. A. Zinov'ev, and N. V. Semakov, "Minimum-check-density codes for correcting bytes of errors, erasures or deffects", *Problems Inform. Transm.*, 19:197-204, 1981.