# Product Perfect Codes and Steganography [1]

H. Rifà-Pous [a], J. Rifà [b,*]

[a] *Department of Computer Science and Multimedia, Universitat Oberta de Catalunya, Barcelona, Spain.*

[b] *Department of Information and Communications Engineering , Universitat Autònoma de Barcelona, 08193-Cerdanyola del Vallès, Spain*

## Abstract

A new coding technique to be used in steganography is evaluated. The performance of this new technique is computed and comparisons with the well-known theoretical upper bound, Hamming upper bound and basic LSB are established.

*Key words:* Embedding rate, encoding, LSB embedding, perfect codes, steganography.

## 1 Introduction and preliminary results

Steganography is an specific application of general data hiding codification with the requirement that the hidden data is undetectable. For instance, we can hide information in an existing file, such as a JPG image file or a MP3 audio file.

An early and easy way to hide information in JPG images is by using LSB (Least Significant Bit) steganography. After performing a discrete cosine or a wavelet transformation in the spatial image followed by a quantization, we can represent an image as a sequence of integer-valued symbols, where each symbol could be represented by $B$ binary digits and we consider only the least

[*] Corresponding author.
*Email address:* `josep.rifa@uab.cat` (J. Rifà).

[1] This work was partially supported by the Spanish MEC and the European FEDER grant MTM2006-03250; the UAB grant PNL2006-13; the MCYT grant PROPRIETAS-WIRELESS SEG2004-04352-C04-04 and the CONSOLIDER CSD2007-00004 ARES project.

significant bit in each symbol (or more than that depending on the protocol we use). By modifying these LSB bits we can hide information in the JPG file. This technique can be viewed as hiding two bits per changed bit in the cover message because, in a random case, 50% bits do not need to be changed (see the “Basic LSB” line in Fig. 1).

A more interesting method, called matrix encoding, is described in [3]. Later, in [8], based on this method, the F5 algorithm is presented which can embed $t$ bits of message in $2^t - 1$ cover symbols by changing, at most, one of them.

The previous method is based on binary linear codes and we can describe it in a general way. We will follow [7] to provide some definitions. Let $n$ and $t$ be positive integers, $t \leq n$, and let $X$ be a finite set. An embedding/retrieval steganographic protocol of type $[n, t]$ over $X$ is a pair of maps $e : X^t \times X^n \longrightarrow X^n$ and $r : X^n \longrightarrow X^t$ such that $r(e(s, v)) = s$ for all $s \in X^t$ and $v \in X^n$. Maps $e$ and $r$ are the embedding and the retrieval maps, respectively. The number $\rho = max\{d(v, e(s, v))|\, s \in X^t, v \in X^n\}$, $d$ being the Hamming distance, is the radius of the protocol.

The embedding map of a $[n, t]$ embedding/retrieval steganographic protocol with radius $\rho$ (for short, a $[n, t, \rho]$ protocol) allows us to hide $t$ information symbols into a string of $n$ cover symbols, by changing a maximum of $\rho$ of these cover symbols.

An important kind of steganographic protocols can be defined from coding theory. Error-correcting codes are commonly used for detecting and correcting errors, or erasures, in data transmission. An explicit description of the relations between error-correcting codes and steganographic systems was presented by Zhang and Li in [9] and shows that there is a corresponding relation between the maximum length embeddable (MLE) codes and perfect error correcting codes.

The most used codes in steganography are linear. The existence of a parity check matrix helps on designing good steganographic protocols.

Let $C$ be a linear $[n, n - t]$ code over the finite field $GF(q)$, equivalently, a linear subspace of $GF(q)^n$ where the dimension is $k = n - t$. The covering radius $\rho$ of code $C$ is defined as $\rho = \max_{\mathbf{v} \in X^n}\{d(\mathbf{v}, C)\}$, where $d(\mathbf{v}, C)$ means the minimum Hamming distance from vector $\mathbf{v}$ to the code $C$. Let $\mathbf{v} = (v_1, v_2, \ldots, v_n) \in X^n$ and define the support of vector $\mathbf{v}$ as the set $supp(\mathbf{v}) = \{\ i \mid x_i \neq 0\ \}$.

Let $X = GF(q)$ and $H$ be a parity check matrix of $C$. The syndrome of any $\mathbf{v} \in X^n$ is the vector $r(\mathbf{v}) = H{\cdot}\mathbf{v}^T$, where $\mathbf{v}^T$ means vector $\mathbf{v}$ as a column vector. The coset $C + \mathbf{v}$ is the set of all vectors in $X^n$ of the same syndrome. A vector $\mathbf{l}_{r(\mathbf{v})}$ of the minimum weight in $C + \mathbf{v}$ will be called leader of the class, although it is not necessarily unique.

The above syndrome map $r : X^n \longrightarrow X^t$, such that $r(\mathbf{v}) = H{\cdot}v^T$ is called the retrieval map of a $[n, t, \rho]$ steganographic protocol, which will be called linear to emphasize that the retrieval map $r$ is a linear map.

The embedding algorithm to compute $e(\mathbf{s}, \mathbf{v})$ for a linear steganographic protocol works in the following way [7,9]:

**Algorithm 1 (Coset Algorithm)** *:*

*(1) compute* $\mathbf{u} := r(\mathbf{v}) - \mathbf{s}$*,*
*(2) define* $e(\mathbf{s}, \mathbf{v}) := \mathbf{v} - \mathbf{l_u}$*, where* $\mathbf{l_u}$ *is a leader of the class of all the vectors in* $X^n$ *with the same syndrome* $\mathbf{u}$*. So,* $r(\mathbf{l_u}) = \mathbf{u}$*.*

We can formulate the above algorithm as a table (leader-syndrome table) consisting of $q^t$ vectors $\mathbf{l_u}$ of length $n$ and associate to each one of them the corresponding syndrome value $r(\mathbf{l_u}) = \mathbf{u}$. Note that the above mentioned F5-Algorithm is an specific case of Algorithm 1, when it is used the Hamming code.

It is easy to check that, after using Algorithm 1, for all $\mathbf{s} \in X^t$ and $\mathbf{v} \in X^n$, we have $r(e(\mathbf{s}, \mathbf{v})) = r(\mathbf{v} - \mathbf{l_u}) = r(\mathbf{v}) - r(\mathbf{l_u}) = \mathbf{u} + \mathbf{s} - \mathbf{u} = \mathbf{s}$. The weight of vector $\mathbf{l_u}$ is the number of changes introduced to vector $\mathbf{v}$. We are interested in computing the average number of these changes.

Let $a_i$, where $0 \leq i \leq \rho$, be the number of vectors of weight $i$ in the constructed table of leaders. Then the average number of embedding changes of the protocol is

$$R_a = \frac{1}{q^t} \sum_{i=1}^{\rho} a_i{\cdot}i. \qquad (1)$$

When code $C$ is the Hamming code the above algorithm and the retrieval function $r$ coincides with the algorithm $F5$ developed in [8]. It is straightforward to see that, in this case, the computational cost of the above algorithm is $O(n)$.

The covering radius $\rho$ of a code is an important parameter in steganography which leads us to consider covering codes. The definition of covering codes and the relations between covering codes and steganography were studied in [1], where it is described the good performance of some covering codes used in steganography.

There are two parameters which help to evaluate the performance of a steganographic protocol $[n, t, \rho]$, the average distortion $D = \dfrac{R_a}{n}$, where $R_a$ is the expected number of changes over uniformly distributed messages and the embedding rate $\frac{t}{n}$. In general, for the same embedding rate a protocol is better when the average distortion is smaller.

In this paper we will focus on the steganographic protocols dealing with the so called "passive warden" case, that is, the case where there is no an adversary modifying the embedded bits. In [6] it is proved that a theoretical hiding capacity exists. It is the supremum of all achievable embedding rates of steganographic protocols subject to a given distortion $D$ under the condition of zero probability of error (that is, there are no changed bits other than those changed by the steganographic protocol). In general, it is hard to compute the hiding capacity, but in some cases this computation can be achieved. Consider, for instance, the case of Bernoulli($\frac{1}{2}$), i.e. the set of cover symbols is $X = \{0, 1\}$ and the sequence of these symbols satisfies the distribution of Bernoulli with $p = \frac{1}{2}$. Let $D$ be the average distortion, then the hiding capacity $C(D)$ for this case (see the "Theoretical bound" line in Fig. 1) has been given by [6]:

$$C(D) = H(D) = -D log_2(D) - (1 - D) log_2(1 - D), \qquad (2)$$

where $0 \leq D \leq 1/2$ and $H$ is the entropy function.

It is important to note that the covering radius $\rho$ of a code $C$ is the largest number of possible changes and the purpose of the embedding function $e(\mathbf{s}, \mathbf{x})$ is to minimize the average number of embedding changes $R_a$. Given a code of length $n$, dimension $k = n - t$ and covering radius $\rho$ we can always compose the conventional steganographic protocol by using the coset algorithm (Algorithm 1) with an embedding rate $k/n$ and average distortion bounded by $\rho/n$. However, as it is pointed out in [4], for the same embedding rate, the smallest average distortion is not always obtained with the code of smallest covering radius. The average distortion is determined by the encoding algorithm. It is the goal of the present paper. To design a steganographic protocol, based on linear codes, such that the average distortion is better than the the average distortion obtained using Algorithm 1.

Our protocol is based on the parity check matrix and while we use the binary Hamming code, that is, the binary linear 1-perfect code, it may be observed that the same ideas apply in other binary perfect codes. For instance the binary Golay code is a good candidate and also the well-known non linear 1-perfect codes but additive codes [2,5] which have a kind of parity check matrix like in the linear binary case.

## 2 Product perfect codes

In this paper we present a highly efficient steganographic protocol. It is based on Hamming codes and both, the embedding and the retrieval algorithm, have the same computational cost as in the F5 case.

As shown in [8], steganographic protocols based on Hamming codes work highly efficiently. Remember that for a Hamming code $[n, n-t]$, where $n = 2^t - 1$, it is easy to compute the embedding rate $t/n = \frac{t}{2^t-1}$ and the average distortion $R_a/n = 1/2^t$ (see (1), where $\rho = 1$ and $a_1 = n$).

Note that the parameters for the average distortion $D = R_a/n$ must be chosen in a narrow range, just the inverses of the different powers of two. To find a protocol with a different average distortion $D$ to the one given by a Hamming code we can easily extend the protocol by using two Hamming codes. Given $D$ we can always find two Hamming codes with average distortions $\dfrac{1}{2^t}$ and $\dfrac{1}{2^{t+1}}$ such that $D = \gamma\cdot\dfrac{1}{2^t} + (1-\gamma)\cdot\dfrac{1}{2^{t+1}}$, where $0 \leq \gamma \leq 1$. We can use both Hamming codes, in the proportion given by $\gamma$, to hide information. The average distortion is the given $D$ and the embedding rate of the obtained protocol is $\gamma\cdot\dfrac{t}{2^t-1} + (1-\gamma)\cdot\dfrac{t+1}{2^{t+1}-1}$ (see the line "Basic F5" in Fig. 1).

Using the well known F5 algorithm as a base, the main idea of the present paper is to use a product code of two Hamming codes with the goal of improving the embedding rate.

Let $C_1$ and $C_2$ be two binary linear codes of length $n_1$ and $n_2$, respectively. The product code $C_1 \otimes C_2$ is the tensor product of the two linear codes $C_1$ and $C_2$. The tensor product is generated by the vectors of the form

$$\mathbf{u} \otimes \mathbf{v} = (u_i v_j \,|\, 1 \leq i \leq n_1,\ 1 \leq j \leq n_2)$$

where $\mathbf{u} = (u_1, u_2, \cdots, u_{n_1}) \in C_1$ and $\mathbf{v} = (v_1, v_2, \cdots, v_{n_2}) \in C_2$.

By the above definition, if we view $\mathbf{u} \otimes \mathbf{v}$ as an $n_1 \times n_2$ matrix with the $i$th row $(u_i v_1, u_i v_1, \cdots, u_i v_{n_2})$, then $C_1 \otimes C_2$ can be viewed as the set of matrices in which every row is an element in $C_2$ and every column is an element in $C_1$.

For the specific case of two Hamming codes, $C_1$ of length $n_1 = 2^x - 1$ and $C_2$ of length $n_2 = 2^y - 1$, the product code $C_1 \otimes C_2$ is the code of length $n = (2^x-1)(2^y-1)$, dimension $k = n - t = (n_1 - x)(n_2 - y)$ and with the peculiarity that their codewords can be seen as $(2^y - 1) \times (2^x - 1)$ matrices, where the rows are codewords in $C_1$ and the columns are codewords in $C_2$.

The steganographic technique we propose in this paper uses product Hamming codes, but does not use Algorithm 1 in the whole product code. Constituent perfect codes of the product code are used in such a way that it it is possible to embed and retrieve hidden bits in a cover source obtaining smaller average distortion $R_a/n$ than that obtained using only perfect codes with the same embedding rate $t/n$.

The protocol that we present is based on three remarkable properties given as lemmas:

**Lemma 1** *Let $C$ be a Hamming code of length $2^t-1$. Let $i,j \in \{1,\ldots,2^t-1\}$. Then there exist a unique coordinate $g_1(i,j) \in \{1,\ldots,2^t-1\}$ such that the vector with support $\{i,j,g_1(i,j)\}$ belongs to $C$.*

*Proof:* It is straightforward since $C$ is a perfect code. □

**Lemma 2** *Let $C$ be a Hamming code of length $2^t-1$. It is always possible to take a parity check matrix such that for any coordinate $1 \leq i \leq 2^{t-1}-1$ there exist two coordinates, specifically the $(2^{t-1}-1+i)$th and the $(2^t-1)$th such that vector $\mathbf{v}$ with support $\{i, g_2(i), 2^t-1\}$ belongs to the code $C$, where $g_2(i) = 2^{t-1}-1+i$.*

*Proof:* Recursively, take the parity check $(t \times 2^t-1)$ matrix $H_t$ for the code of length $2^t-1$:

$$H_t = \left(\begin{array}{c|c|c} 0\cdots 0 & 1 \;\; \cdots \;\; 1 & 1 \\ \hline H_{t-1} & H_{t-1} & 0 \end{array}\right),$$

where we can begin the sequence with the parity check matrix for a code of length $2^2-1$:

$$H_2 = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

□

**Lemma 3** *Let $C$ be a Hamming code of length $n = 2^t-1$ with parity check matrix $H$ as in the above Lemma 2. Let $i,j,r \in \{1,\ldots,2^{t-1}-1\}$ and $i<j<r$ such that vector $\mathbf{u}$ with support $\{i,j,r\}$ belongs to code $C$. Then, the vector $\mathbf{v}$ with support $\{j, g_2(i), g_2(r)\}$ belongs to code $C$ and $g_2(i), g_2(r) \in \{2^{t-1},\ldots,2^t-2\}$.*

*Proof:* From Lemma 2 vectors $\mathbf{u}_1, \mathbf{u}_2$ with support $\{i, g_2(i), 2^t-1\}$ and $\{r, g_2(r), 2^t-1\}$, respectively, belong to $C$ and $g_2(i), g_2(r) \in \{2^{t-1},\ldots,2^t-2\}$.

Therefore, the vector $\mathbf{v} = \mathbf{u} + \mathbf{u}_1 + \mathbf{u}_2$ belongs to $C$ and has the wanted support. □

Given two Hamming codes $C_1$ and $C_2$ of lengths $2^x-1$ and $2^y-1$, respectively, consider $C_1 \otimes C_2$ and define the following steganographic protocol:

**Algorithm 2**

We will take blocks in the cover source of size $(2^y - 1) \times (2^x - 1)$.

**Embedding:**

**rows:**

In each row $i$ we apply Algorithm 1. There will be a maximum of one bit changed in each row.

After finishing with all the rows we go to process the columns.

**columns:**

Process the columns starting from the first one until the $c$th column, where $c \leq (2^{x-1} - 1)$th column.

Take the $i$th column $(1 \leq i \leq c)$ and apply Algorithm 1 to find a maximum of one coordinate $c_i$ to be changed. When such a coordinate $c_i$ exists it corresponds to a $j$th row already processed as a row.

Change the coordinate $c_i$ and also the coordinates $2^x - 1$ and $2^{x-1} - 1 + c_i$ in the $j$th row (see Lemma 2).

**Retrieval:**

For each row and also for the first $c$ columns the retrieval algorithm is the same as in Algorithm 1.

**Theorem 1** *Algorithm 2 allows to embed $x(2^y - 1) + yc$ bits in any cover source of size $(2^y - 1) \times (2^x - 1)$.*

*Proof:* It is straightforward for the embedding of $x(2^y - 1)$ bits in the rows. For the embedding of $y$ bits in any of the first $c$ columns $(1 \leq c \leq 2^{x-1} - 1)$ columns, note that given a vector of length $2^x - 1$ it is possible to change the value of one of the first $2^{x-1} - 1$ coordinates and two more of the last $2^{x-1}$ coordinates (see Lemma 2) without varying the value of the syndrome. This result assure that when we process the first $c$ columns in Algorithm 2 we can embed $y$ bits in each one of these columns without disturbing the previously processed rows or columns. □

It is possible to slightly modify the above Algorithm 2 to obtain lower average distortion for the same embedding rate. In processing the columns we can use Lemmas 1, 2 and 3 to adjust the average distortion as we propose in the following algorithm where we described with great detail all the possible situations to show that it works in all the possible cases.

**Algorithm 3 (Product perfect codes)**

We will take blocks in the cover source of size $(2^y - 1) \times (2^x - 1)$.

**Embedding:**

**rows:**

In each row $i$ we will apply Algorithm 1 and we will change a maximum of one bit. Say $r_i$ the coordinate where the changed bit in the $i$th row lies. The expected number of changes in each row could be computed from (1) as $R_a = \frac{1}{2^x}\sum_{i=1}^{\rho}\binom{2^x-1}{i}\cdot i = \frac{2^x-1}{2^x}$.

After finishing with all the rows we go to the columns.

**columns:**

Process the columns starting from the first column until the $c$th column, where $1 \leq c \leq (2^{x-1}-1)$th column.

Take the $i$th column ($1 \leq i \leq c$) and apply Algorithm 1 to find a maximum of one coordinate $c_i$ to be changed. When such a coordinate $c_i$ exists it corresponds to a $j$th row which was already processed as a row.

(1) Coordinate $c_i$ coincides with $r_j$ (obtained when we previously processed the $j$th row). In such case we change the coordinate $c_i = r_j$ and, to avoid non desirable effects in the $j$th row, we will also change the two coordinates $2^{x-1}-1+i$ and $2^x-1$ as is stated in Lemma 2. This situation increases by one the amount of embedded bits and happens in $\frac{1}{2^x}$ occasions.

(2) Coordinate $c_i$ corresponds to the $j$th row. In this $j$th row there is no any previously processed $r_j$. In such case we change the coordinate $c_i$ and, to avoid non desirable effects in the $j$th row, we will also change the two coordinates $2^{x-1}-1+i$ and $2^x-1$ as is stated in Lemma 2. This situation increases by three the amount of embedded bits and happens in $\frac{1}{2^x}$ occasions.

(3) Coordinate $c_i$ corresponds to the $j$th row. In the $j$th row there is a previously processed $r_j < c_i$. In such case we change the coordinate $c_i$ and, to avoid non desirable effects in the $j$th row, we will also change the two coordinates $2^{x-1}-1+i$ and $2^x-1$ as is stated in Lemma 2. This situation increases by three the amount of embedded bits and happens in $\frac{i-1}{2^x}$ occasions.

(4) Coordinate $c_i$ corresponds to the $j$th row. In the $j$th row there is a previously processed $r_j > c_i$ with $g_1(r_j, c_i) > c_i$ (see Lemma 1). In such case we change the coordinate $c_i$ and, to avoid non desirable effects in the $j$th row, we also change the coordinate $r_j$ and $g_1(r_j, c_i)$. This situation increases by one the amount of embedded bits and happens at least in $\frac{2^x-2i}{2^x}$ occasions.

(5) Coordinate $c_i$ corresponds to the $j$th row. In the $j$th row there is a previously processed $r_j > c_i$ with $g_1(r_j, c_i) < c_i$ (see Lemma 1). In such case we change the coordinate $c_i$ and, to avoid non desirable effects in the $j$th row, we will also change the coordinate $r_j$ and the two coordinates $g_2(c_i)$ and $g_2(r_j)$ as is stated in Lemma 3. This situation increases by two the amount of embedded bits and happens at most in $\frac{i-1}{2^x}$ occasions.

**Retrieval:**

For each row and also for the first $c$ columns the retrieval algorithm is the same as in Algorithm 1.

After processing all the rows, and before beginning with the columns, the average distortion is $R_a/(2^x-1)(2^y-1)$, where:

$$R_a = \frac{(2^y-1)(2^x-1)}{2^x}$$

and the embedding rate:

$$\frac{x(2^y-1)}{(2^x-1)(2^y-1)}$$

As we stated in the algorithm, each processed column $i$ adds, at least, the following fraction to the expected number of changes:

$$\Big(\frac{2^y-1}{2^y}\Big)\Big(\frac{1}{2^x}\cdot 1+\frac{1}{2^x}\cdot 3+\frac{i-1}{2^x}\cdot 3+\frac{2^x-2i}{2^x}\cdot 1+\frac{i-1}{2^x}\cdot 2\Big)=\Big(\frac{2^y-1}{2^y}\Big)\Big(\frac{2^x-3i-1}{2^x}\Big),$$

and, summing up, we can deduce the following formulas for the expected number of changes, after processing all the rows and the first $c$ columns ($1 \leq c \leq 2^{t-1}-1$):

$$\begin{aligned} R_a =& \frac{(2^y-1)(2^x-1)}{2^x}+\frac{2^y-1}{2^y}\sum_{i=1}^{c}\frac{2^x+3i-1}{2^x}= \\ & \frac{(2^y-1)(2^x-1)}{2^x}+\frac{(2^y-1)(2^{x+1}+3c+1)c}{2^{x+1}2^y}, \end{aligned}$$

the average distortion is $R_a/(2^x-1)(2^y-1)$ and the embedding rate:

$$\frac{x(2^y-1)+yc}{(2^x-1)(2^y-1)}.$$

The performance of Algorithm 3 is shown in Fig. 1. For a fixed value of $x$ and $y$ it is showed the performance for the all the values $1 \leq c \leq 2^{x-1}-1$. Almost all of them are better than the corresponding codes with the same average distortion and using the standard Algorithm 1.

In all these product codes we have $2^y-1$ rows and $2^x-1$ columns. The proposed algorithm works better (the embedding rate is higher for the same average distortion) when $y$ is greater than or equal to $x$, but as $x$ and $y$ differ, the results are poorer. Hence, the best results we obtain are for $x=y$.

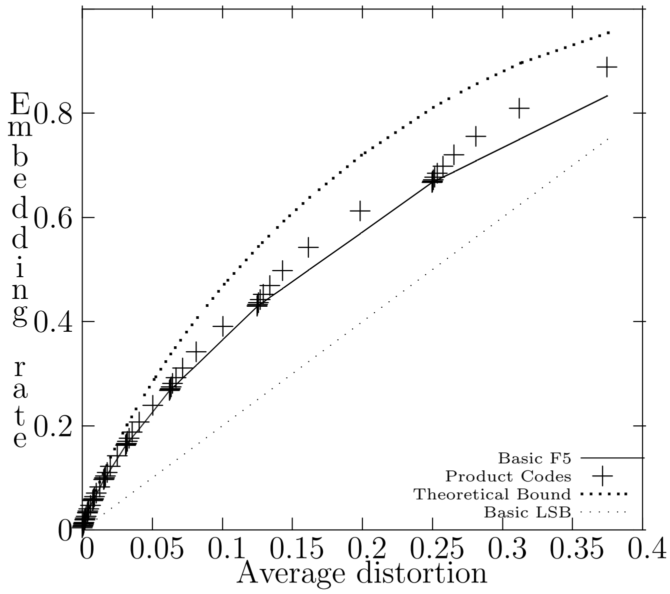

Fig. 1. Performance of steganographic codes. The embedding rate as a function of the average distortion.

## 3 Conclusions

Using a new technique that is different from the usual Algorithm 1 we obtain a steganographic protocol for embedding hidden information in a cover source, and which performs better. As it is shown in Fig. 1 we conclude that with the proposed Algorithm 3 we obtain better performance than that obtained using basic LSB steganography or the basic F5 algorithm.

## References

[1] J. Bierbrauer and J. Fridrich, Constructing good covering codes for applications in steganography, Available at http://www.math.mtu.edu/∼jbierbra/ (2006).

[2] J. Borges and J. Rifà, “A characterization of 1-perfect additive codes”, *IEEE Trans. Information Theory*, vol. 45(5), pp. 1688-1697, 1999.

[3] R. Crandall, Some notes on steganography, (1998).

Available at http://os.inf.tu-dresden.de/∼westfeld/crandall.pdf.

[4] J. Fridrich, P. Lisoněk, D. Soukal, “On steganographic embedding efficiency”, Springer-Verlag, LNCS 4437, pp. 282-296, 2007.

[5] D.S. Krotov, “$\mathbb{Z}_4$-linear Hadamard and extended perfect codes”, *Proc. of the International Workshop on Coding and Cryptography*, Paris (France), Jan. 8-12, pp. 329–334, 2001.

[6] P. Moulin, Y. Wang, New results on steganographic capacity, Proceeding of CISS 2004. University of Princeton, Princeton, New Jersey (2004) Available: http://www.ifp.uiuc.edu/%7Eywang11/paper/CISS04_204.pdf

[7] C. Munuera, "Steganography and error-correcting codes", Signal Processing 87, pp. 1528–1533, 2007. Available online at www.sciencedirect.com.

[8] A. Westfeld, High Capacity Despite Better Steganalysis (F5 - A Steganographic Algorithm), Springer-Verlag, LNCS 2137, pp. 289-302, 2001.

[9] W. Zhang and S. Li, "A Coding Problem in Steganography". To appear in Designs, Codes and Cryptography. A preprint available at http://arxiv.org/abs/cs/0505072.