8-Rooks Solutions for Image Steganography Technique

Manjot Kaur Bhatia
Jagan Institute of Management Studies, New Delhi, India

Exchange of secret messages on the internet is a big issue for the researchers. This raises the need of an information hiding technique, Steganography is an art of hiding digital information in another digital media used as cover. The steganographic method based on Least Significant Bit(LSB) insertion approach provides high degree of visual quality and increases the capacity of the embedded data but secret data is not highly secured by this method. In this paper we proposed a message hiding technique that randomizes the pixel selection for hiding message bits in an image, based on solutions of 8-Rooks problem of placing 8-non-attacking Rooks on an 8*8 chessboard. The cover image is divided into 8*8 pixel blocks and pixels are selected from each block corresponding to the positions of 8-rooks in the different solutions for 8-rooks problem. LSB of the above selected pixels are collected and compared with ASCII code of all the characters in the secret message. Index number of the character matched with ASCII code of collected bits is embedded into the LSB of the last row of an 8*8 pixel block. The experimental results show that the proposed algorithm improves the security, robustness and embedding capacity.

Keywords: Image Steganography, 8-Rooks Problem, LSB embedding technique, Information Hiding.

1. INTRODUCTION

In todays internet world, protection and security of transmitted data is a serious issue for the researchers. This leads to the demand of a strong data security method. Steganography and Cryptography are both intended to protect information from unwanted parties. In comparison to cryptography which makes data unreadable for the third party, data hiding scheme hides the existence of data. It is for this reason that most experts would suggest using both to add multiple layers of security. Data hiding scheme is used to conceal the secret information in cover media so that no one can predict the presence of secret information. Steganography, a branch of information security, is the technique of hiding digital information in another digital media in such a way that it should not make any visible changes in the cover media. The digital media used to hide the secret information is called cover object or host object. A secret message is embedded into the cover medium. The cover media and secret information can be image, video or audio file. The most popular formats used are .bmp, .doc, .gif, .jpeg, .mp3, .txt and .wav.. These formats are also popular because of the relative ease by which redundant or noisy data can be removed from them and replaced with a hidden message. Any key used to embed and extract the secret message is called stego key. A stego object is obtained after embedding secret message into cover object, which modifies the cover object. When a large amount of data is embedded into an image the visual specifications of image such as colour and smoothness are altered (Anderson, R. J. et.al., 1998).The important factors that need to be considered in image steganography are imperceptibility, robustness and capacity of the hidden data. It is important to use the strong embedding method so that the attacker should not be able to guess the extraction mechanism. Steganogrphic methods could be categorized into spatial domain embedding and frequency domain embedding. The first type utilizes the spatial domain of a cover image to hide the secret data into the pixels of the cover image. The Least Significant Bit (LSB) insertion (Chan, C. K et. al., 2004 and Dey S. et al., 2007) is the most common spatial domain technique (Daneshkhah, A. et al., 2011) which consecutively replaces the least significant bit of cover image with the message bits. LSB method take advantage of the natural weakness of Human Visual System (HVS) (Yi-zhen, C. et al.,2010) in recognizing the slight difference of colours. In this method least significant n-bits of target pixel in cover image are replaced with message bits [4-21]. Some researchers used the idea of sensitiveness of human eye towards smooth areas as compare to edge areas and embedded additional bits in edge area than smooth area(Padmaa, M., et. al., 2011,Chan, CK. et. al., 2004, Thien, CC. et. al., 2003, Yang, CH., 2008). Wang(2008), Wu and Tsai(2003,2005) proposed a modified LSB method based on pixel-value differencing approach that improves and determines the embedding capacity. Pfitzmann and Westfield introduced a method based on statistical analysis of Pairs of Values (PoVs) that are exchanged during message embedding. Chan and Cheng [2004] proposed a simple LSB substitution along with optimal pixel adjustment process (OPAP). Amirtharajan and Balaguru (2009) presented a comparative analysis of various image steganographic methods. In frequency domain, steganographic methods are based on Discrete Cosine Transformation (DCT) or Discrete Wavelet Transformation (DWT).In DCT, after performing DCT on image blocks and quantizing the DCT coefficients, message bits are embedded into the quantized DCT coefficients. In this paper we have proposed secret message hiding technique using solutions of the problem of placing non-attacking Rooks in a chessboard and ISAAC encryption algorithm. Problem of placing rooks at non-attacking positions in a chessboard is explained as:

a) **Rook polynomial in a chessboard:** The theory of rook polynomials was introduced by Kaplansky and Riordan (Kaplansky, I. and Riordan, J, 1946) and developed further by Riordan(Riordan, J., 2002). A rook polynomial provides different ways to set non-attacking rooks on a chessboard. The rook polynomial $R_B(x)$ is a function that generates the number of ways of arranging k non-attacking rooks on a board of B.

$$R_B(x) = \sum_{k=0}^{\infty} r_k(B)x^k \tag{1}$$

where r_k represents the number of ways to set k non-attacking rooks on the board(Wikipedia, December,2014). The rooks are positioned in such a way on the board that no two rooks should be in the same row or column. Hence, this arrangement is placing the rooks on a static, fixed board. Thus arrangement will not change if the chessboard is turned around with the squares remain fixed. The polynomial will remain same if rows are interchanged or columns are interchanged (Wikipedia, December, 2014). There is fixed value for the maximum number of non attacking rooks that can be placed in on board with fixed no. of rows and columns. The board cannot adjust rooks more than the smaller number from number of rows or number of columns. According to rook polynomial given by Kaplansky and Riordan, rook polynomials for few square boards are represented as:

$$R_1(x) = x + 1 \tag{2}$$

$$R_2(x) = 2x^2 + 4x + 1 \tag{3}$$

$$R_3(x) = 9x^3 + 18x^2 + 9x + 1 \tag{4}$$

$$R_4(x) = 24x^4 + 96x^3 + 72x^2 + 16x + 1 \tag{5}$$

From the above given rook polynomials, equation no.(5) says that there are 24 ways of placing 4 rooks, 96 ways of placing 3 rooks, 72 ways of placing 2 rooks, 16 ways of placing 1 rook and 1 way of placing zero rook on a square board of 4*4. The basic idea behind this rook polynomial is ”Eight rooks problem” by H. E. Dudeney (Dudeney, H., E.,1917). Dudeney shows that maximum eight non attacking rooks can be placed on an 8*8 chessboard and the problem is to find the number of ways of arranging 8 rooks on 8*8 chessboard at non-attacking positions. Vilenkin gave the solution to the above problem with equation no. 6, to calculate the total number of possible non-attacking rooks arrangements (Vilenkin, N.,Ya.,1969) in m*n chessboard:

$$r_k = \binom{m}{k}\binom{n}{k}k! = \frac{n!m!}{k!(n-k)!(m-k)!} \tag{6}$$

According to equation no. 6, total number of possible ways of arranging 8-non-attacking rooks in an 8*8 chessboard is 8!.

b) **ISAAC:** ISAAC(indirection, shift, accumulate, add, and count) is a pseudorandom number generator and a stream cipher(Robert J., 1996). It is a cryptographically secure pseudorandom number generator designed by Robert J. Jenkins Jr. in 1996.

In this paper we have proposed a message hiding technique based on the problem of placing 8-rooks at non-attacking positions in an 8*8 chessboard. The proposed technique decomposes the cover image into three image plane. Each image is further divided into 8*8 pixels blocks. The different arrangements showing positions of 8-rooks on 8*8 chessboard are matched with 8*8 pixels blocks. The LSB of the first 7-pixels corresponding to positions of rooks are selected and matched with 7-bit ASCII code of the all the characters in the message to be embedded. The position of the matched character in the message and of the corresponding solution number of 8-rooks problem is encrypted using ISAAC encryption with randomly generated password. The 8-bit ASCII code of the encrypted value of position of matched character in the message and of the corresponding solution number of 8-rooks problem is embedded into the LSB and 2^{nd} LSB of last row of the image block. If the LSB of the pixels selected from block after matching with all the solutions of 8-rooks does not match with any character in the message, replace LSB of the last row of the block with bit "0" and move to next block. The proposed algorithm is tested on images of different formats such a BMP, PNG, JPG. The cover image and stego image is compared using Peak Signal to Noise ratio(PSNR) and the experimental results of proposed algorithm shows high PSNR . The paper is organised in four sections. In section 2, we present the basics of method used in proposed image steganography method. The proposed embedding and extraction algorithms are presented in the section 3. Experimental results are presented in the section 4. The paper is concluded in the last section.

2. PROPOSED IMAGE STEGANOGRAPHY TECHNIQUE

The objective of our research is to propose an image steganography technique that hides secret data in an image in a way that it needs to do slight modifications in the cover image. The proposed technique divides the cover image into blocks. Each image block is considered as a chessboard and proposed algorithm chooses the pixels based on the positions of placing non-attacking Rooks in a chessboard. In our proposed method a digital image is seperated into R, G and B planes. The RGB planes are considered as separate images. Secret message is divided into three parts. Each image plane is used to hide one part of the message. An image plane is divided into 8*8 byte blocks. In our algorithm, we are using the solution set for the problem of placing 8-non-attacking Rooks in an 8*8 chessboard. The embedding algorithm does not replace the message bits with LSB bits of the targeted pixels. It extracts the LSB of the pixels, located at the positions corresponding to the placement of Rooks in a chessboard. Proposed algorithm then tries to match the ASCII character equivalent of the extracted bits with characters in the message. To enhance the security of the proposed algorithm, ISAAC encryption is used to encrypt the information like: index no. of the matched character from the message and the applied solution number from solution set, using stego key. This encrypted information is then embedded into the LSB of the pixels in the 8th row of the cover image plane and is required to extract the message. When all the message characters are embedded into the blocks of cover image, stego image is generated and transmitted to the receiver. Receiver uses the extraction algorithm and the received stego key to extract the secret message. Figure 1 shows the diagram for the proposed message hiding technique.

Figure 2, shows some of the possible solutions of 8 rooks problem. First solution in Figure 2 shows that the first rook can take the position on any one of eight squares starting from bottom.

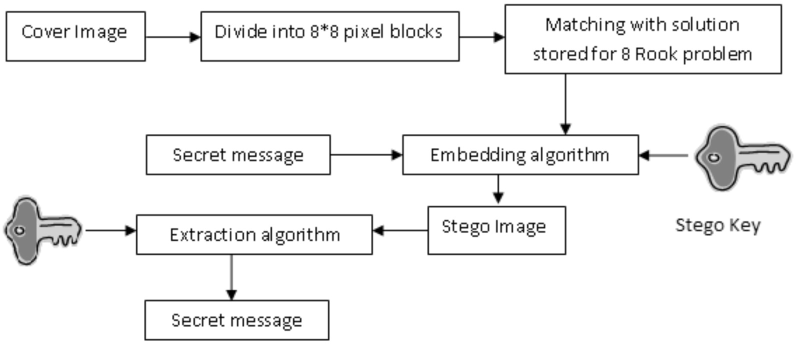


Figure 1: Sequence of operations in the proposed method

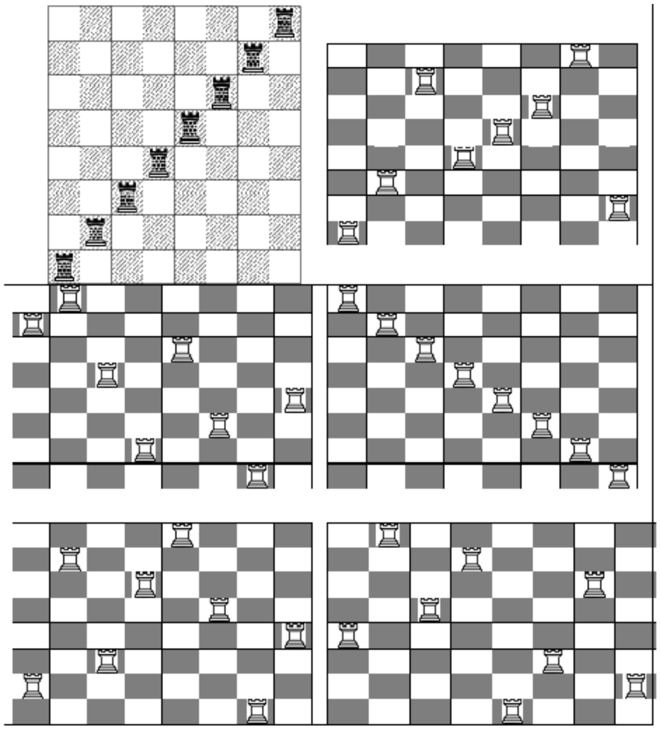


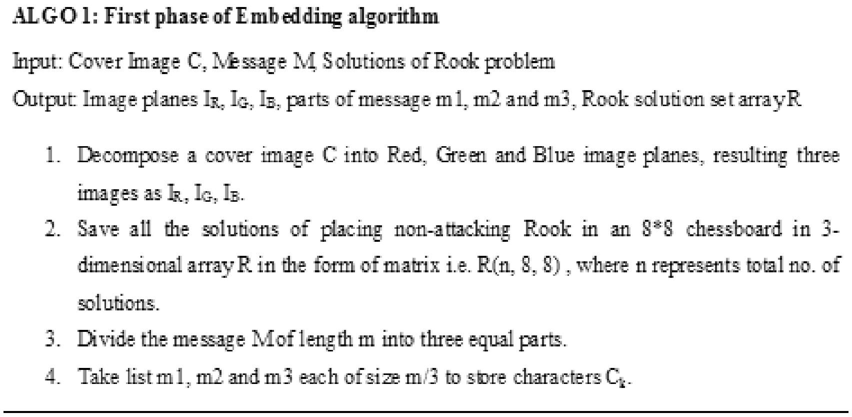
Figure 2: Some of the possible solutions for 8 rooks problem

Second rook has to the choice of seven squares; third rook has the option of six squares, fourth rook five squares, fifth rook four squares, sixth rook three squares and so on. Thus, the number of different methods of placing eight rooks on 8*8 chessboard must be $8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$ = 40,320, i.e. 8!. These 40,320 solutions can be used to hide the secret message into an image.

3. PROPOSED EMBEDDING AND EXTRACTION ALGORITHM

3.1 Embedding algorithm

The proposed algorithm decomposes the cover image into Red, Green and Blue image planes. The algorithm divides the secret message M into three parts m1, m2 and m3. Image plane Red is used to hide m1, Green for m2 and Blue is used for hiding m3. Utilizing each image plane for hiding 1/3 part of the message increases the embedding capacity of the cover image. Each image plane is divided into 8*8 pixel blocks. The embedding algorithm stores the different solutions of placing 8-Rooks in an 8*8 chessboard, some of the possible solutions are shown in Figure 2. The positions of rooks on the chessboard in a solution mark the pixel positions for message hiding in an 8*8 pixel block of an image plane. The proposed embedding algorithm selects pixels from 8*8 pixel block of the image corresponding to positions of 8-Rooks in the solution from stored solution set. Now, collects the LSB of the above selected 7-pixels excluding the pixel of the last row. Convert the selected 7-bits into its equivalent ASCII character and compare it with every character from the message part to be embedded in that image plane. If it matches with any character from the message part, index no. of that character in the message and the applied solution number from the solution set is encrypted with ISAAC encryption with randomly generated secret key. Now, the 8-bits of encrypted value of index no. of matched character and of solution number are embedded into the 2^{nd} LSB and LSB of the 8-pixels of the last row of 8*8 pixel block. If ASCII character equivalent to collected 7-bits does not match with any character in the message, replace LSB of the last row of 8*8 pixel block with bit "0". It marks no character is embedded into this block. Now, move to next 8*8 pixel block and again apply all the stored solutions in the solution set. The steps involved in embedding algorithm are shown Figure 3 and algorithm is given in ALGO 1 and ALGO 2.



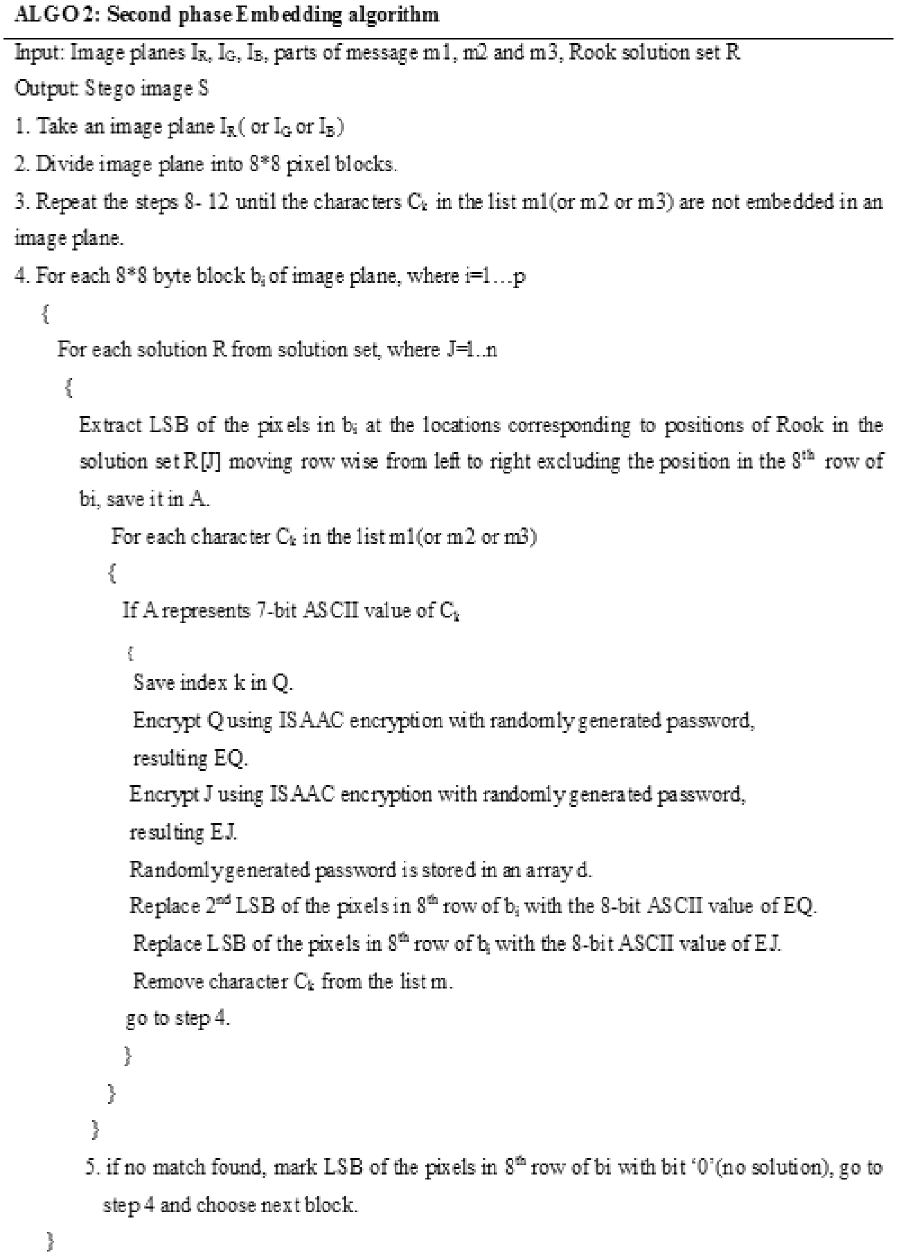
ALGO 1: First phase of Embedding algorithm

Input: Cover Image C, Message M, Solutions of Rook problem

Output: Image planes I_R, I_G, I_B, parts of message m1, m2 and m3, Rook solution set array R

1. Decompose a cover image C into Red, Green and Blue image planes, resulting three images as I_R, I_G, I_B.
2. Save all the solutions of placing non-attacking Rook in an 8*8 chessboard in 3-dimensional array R in the form of matrix i.e. R(n, 8, 8) , where n represents total no. of solutions.
3. Divide the message Mof length m into three equal parts.
4. Take list m1, m2 and m3 each of size m/3 to store characters C_k.

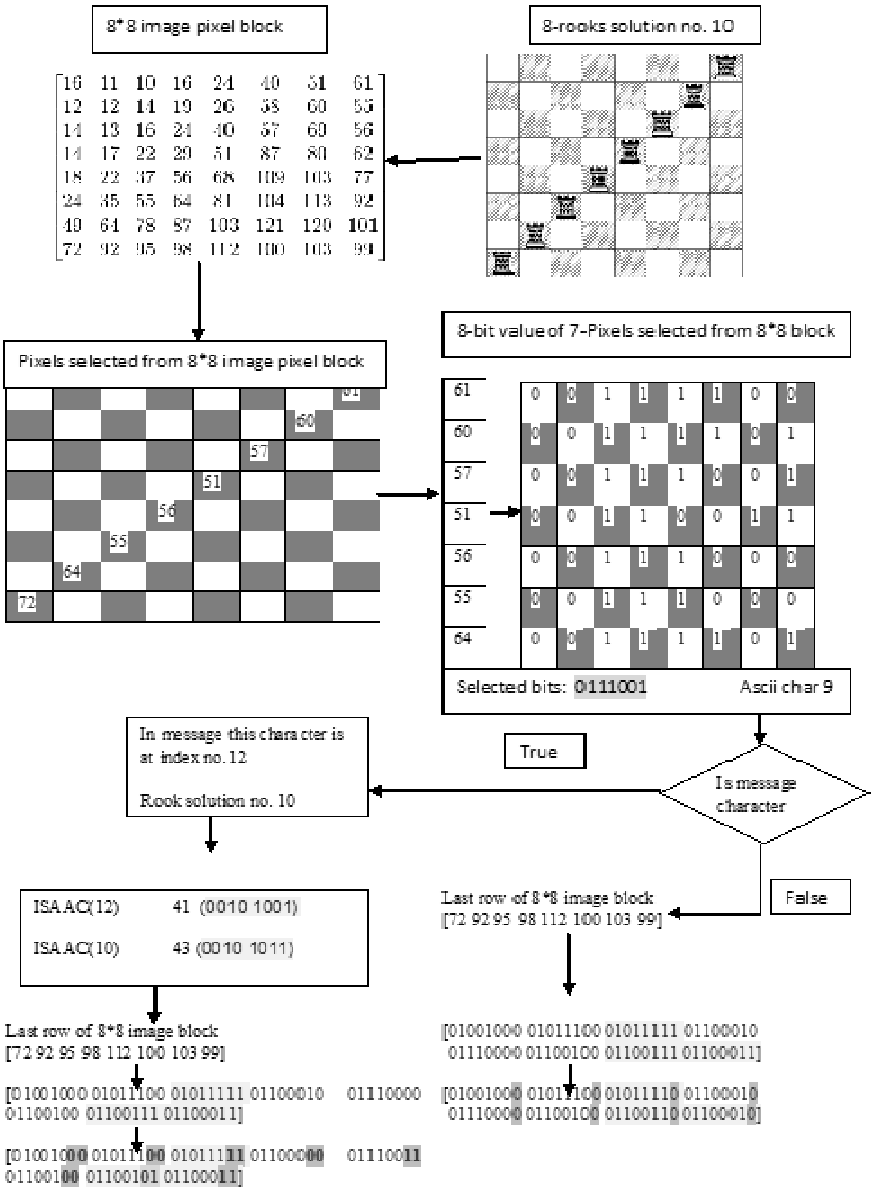
ALGO 1: First phase of Embedding algorithm



ALGO 2: Second phase Embedding algorithm

```
Input: Image planes $I_R$, $I_G$, $I_B$, parts of message m1, m2 and m3, Rook solution set R
Output: Stego image S
1. Take an image plane $I_R$( or $I_G$ or $I_B$)
2. Divide image plane into 8*8 pixel blocks.
3. Repeat the steps 8- 12 until the characters $C_k$ in the list m1(or m2 or m3) are not embedded in an image plane.
4. For each 8*8 byte block $b_i$ of image plane, where i=1…p
  {
    For each solution R from solution set, where J=1..n
     {
       Extract LSB of the pixels in $b_i$ at the locations corresponding to positions of Rook in the
       solution set R[J] moving row wise from left to right excluding the position in the 8th row of
       bi, save it in A.
           For each character $C_k$ in the list m1(or m2 or m3)
           {
             If A represents 7-bit ASCII value of $C_k$
              {
               Save index k in Q.
               Encrypt Q using ISAAC encryption with randomly generated password,
               resulting EQ.
              Encrypt J using ISAAC encryption with randomly generated password,
              resulting EJ.
              Randomly generated password is stored in an array d.
              Replace 2nd LSB of the pixels in 8th row of $b_i$ with the 8-bit ASCII value of EQ.
               Replace LSB of the pixels in 8th row of $b_i$ with the 8-bit ASCII value of EJ.
               Remove character $C_k$ from the list m.
              go to step 4.
              }
           }
        }
     5. if no match found, mark LSB of the pixels in 8th row of bi with bit '0'(no solution), go to
        step 4 and choose next block.
  }
```

ALGO 2: Second phase of Embedding algorithm



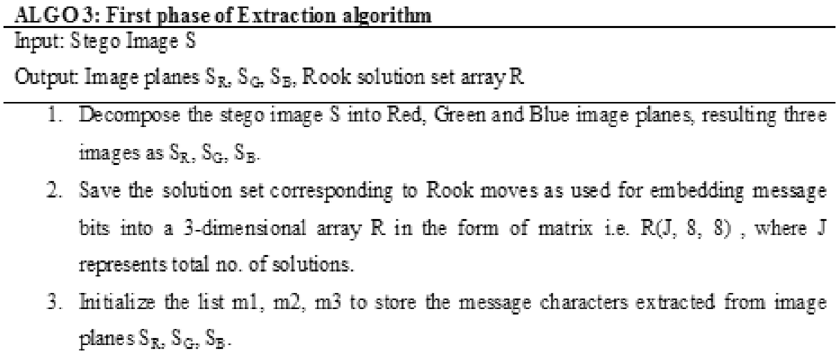
16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

61	0	0	1	1	1	1	0	0
60	0	0	1	1	1	1	0	1
57	0	0	1	1	1	0	0	1
51	0	0	1	1	0	0	1	1
56	0	0	1	1	1	0	0	0
55	0	0	1	1	1	0	0	0
64	0	0	1	1	1	1	0	1

Figure 3: Steps of Embedding algorithm

3.2 Extraction algorithm

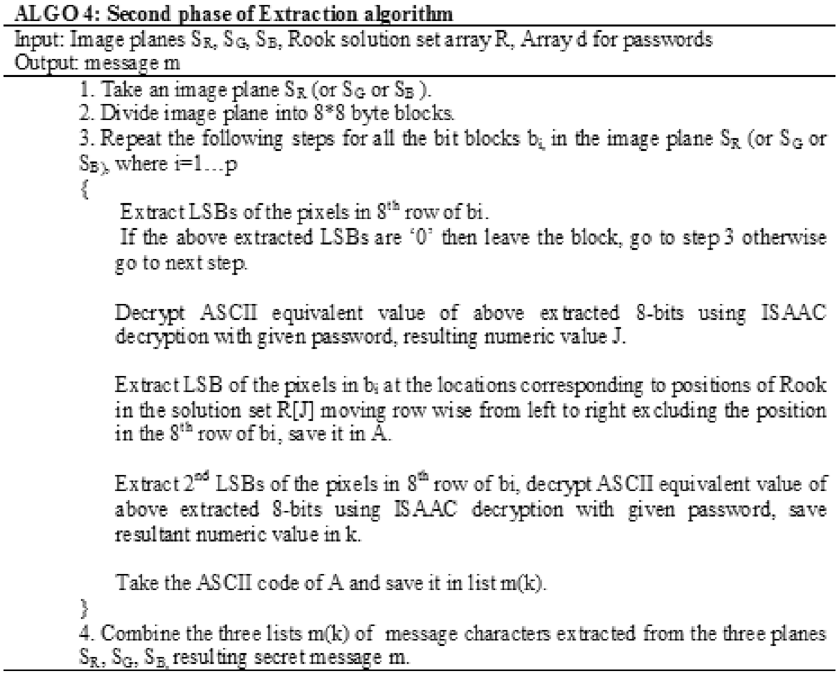
The extraction algorithm is the reverse process of embedding algorithm. The first two steps of extraction algorithm are same as the steps used for embedding message. The stego image is decomposed into Red, Green and Blue image planes. Each image plane is divided into 8*8 pixels blocks. The 7-bits are collected for solution number to be applied to pixel block to extract the message character and for the position of extracted character in the message. For decryption process, receiver uses the same stego key passed to it by the sender. Receiver follows the steps given in the extraction algorithm to extract the secret message. The steps involved in extraction algorithm are given in ALGO 3 and ALGO 4.



```
ALGO 3: First phase of Extraction algorithm
Input: Stego Image S
Output: Image planes $S_R$, $S_G$, $S_B$, Rook solution set array R

    1. Decompose the stego image S into Red, Green and Blue image planes, resulting three
       images as $S_R$, $S_G$, $S_B$.
    2. Save the solution set corresponding to Rook moves as used for embedding message
       bits into a 3-dimensional array R in the form of matrix i.e. R(J, 8, 8) , where J
       represents total no. of solutions.
    3. Initialize the list m1, m2, m3 to store the message characters extracted from image
       planes $S_R$, $S_G$, $S_B$.
```

ALGO 3: First phase of Extraction algorithm



```
ALGO 4: Second phase of Extraction algorithm
Input: Image planes $S_R$, $S_G$, $S_B$, Rook solution set array R, Array d for passwords
Output: message m
    1. Take an image plane $S_R$ (or $S_G$ or $S_B$ ).
    2. Divide image plane into 8*8 byte blocks.
    3. Repeat the following steps for all the bit blocks $b_i$ in the image plane $S_R$ (or $S_G$ or
    $S_B$), where i=1…p
    {
        Extract LSBs of the pixels in 8th row of bi.
        If the above extracted LSBs are '0' then leave the block, go to step 3 otherwise
        go to next step.

        Decrypt ASCII equivalent value of above extracted 8-bits using ISAAC
        decryption with given password, resulting numeric value J.

        Extract LSB of the pixels in $b_i$ at the locations corresponding to positions of Rook
        in the solution set R[J] moving row wise from left to right excluding the position
        in the 8th row of bi, save it in A.

        Extract 2nd LSBs of the pixels in 8th row of bi, decrypt ASCII equivalent value of
        above extracted 8-bits using ISAAC decryption with given password, save
        resultant numeric value in k.

        Take the ASCII code of A and save it in list m(k).
    }
    4. Combine the three lists m(k) of message characters extracted from the three planes
    $S_R$, $S_G$, $S_B$ resulting secret message m.
```

ALGO 4: Second phase of Extraction algorithm

3.3 Experimental Results

Experiments are conducted on the images with different formats like BMP, PNG and JPG. To evaluate the performance of the proposed algorithm, we have performed experiments on the original 256 × 256 images. Six images are taken as cover images, which are listed in Figure 4. Peak-Signal-to-Noise ratio (PSNR) is used to measure the quality of the stego image, which is defined in equation 6, for an $M \times N$ gray scale image.

Where $p_{i,j}$ and $q_{i,j}$ represent the cover image pixels and stego image pixels. M and N represent height and width respectively of the images. Cover image and stego image are considered to be similar with PSNR value greater than 30dB and it is hard to make difference between the cover image and its corresponding stego image through human eyes. Histogram of images is also used to compare the cover image and stego image. Little difference in shape of the histogram of stego image proves that the message is safe. Results obtained after inserting message 4000 to 12000 bits using our proposed algorithm in the images given in Figure 4 are shown in the Figures 5-10. It can be observed from the experimental results that the shape of the histogram is preserved after hiding the message.

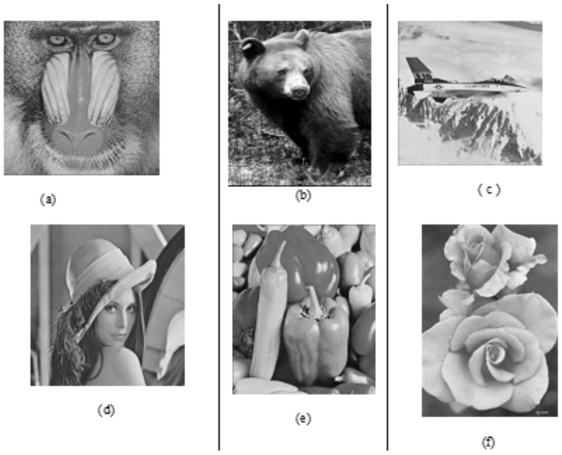


Figure 4: (a) Baboon.jpg (b) Bear.bmp (c) F16.png (d) Lena.jpg (e) Peppers.png (f) brandyrose.bmp

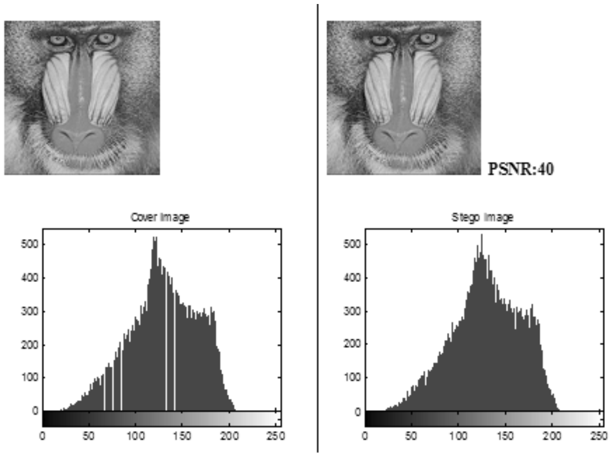


Figure 5. Baboon.jpg (a) cover image (b) Stego image (c) cover image histogram (d) stego image histogram

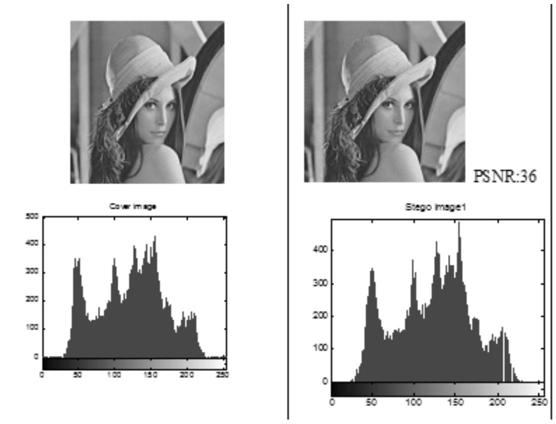


Figure 6. Lena.jpg (a) cover image (b) Stego image (c) cover image histogram (d) stego image histogram

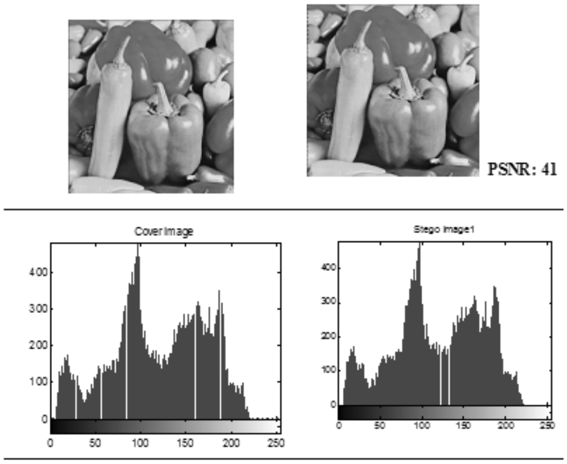


Figure 7. peppers.png (a) cover image (b) Stego image (c) cover image histogram (d) stego image histogram

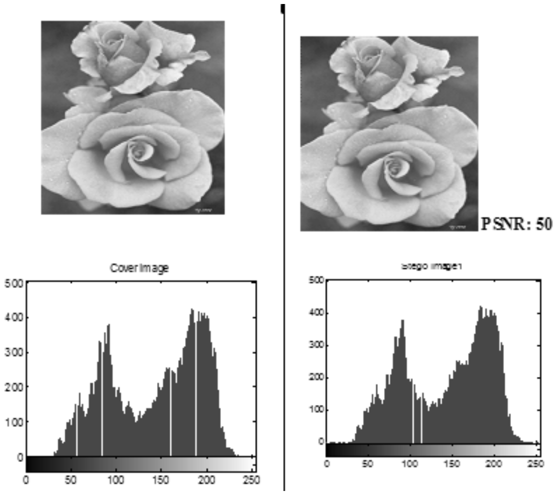


Figure 8. Brandyrose.bmp (a) cover image (b) Stego image (c) cover image histogram (d) stego image histogram

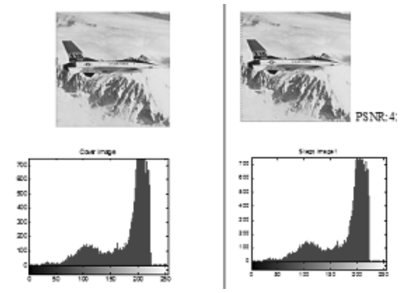


Figure 9. f16.png (a) cover image (b) Stego image (c) cover image histogram (d) stego image histogram

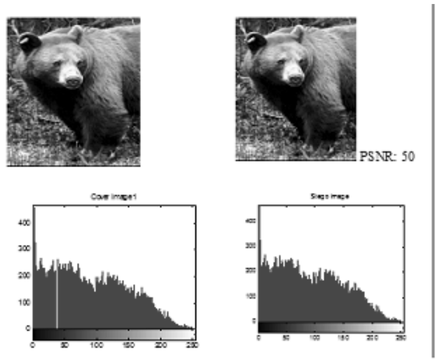


Figure 10. Bear.bmp (a) cover image (b) Stego image (c) cover image histogram (d) stego image histogram

4. CONCLUSION AND FUTURE WORK

The proposed algorithm uses the solutions of placing 8-non attacking Rooks on an 8*8 chessboard for pixel selection from an 8*8 blocks of cover image. The algorithm increases the embedding capacity by using the three planes RGB of the cover image for hiding message. Proposed embedding method uses the LSB of the selected pixels and compares it with the message bits. The embedding method tries to match the LSB of the selected pixels with message bits rather than replacing it with the message bits. It improves the quality of the stego image. Some hidden information is encrypted with ISAAC using stego key. The proposed method is secure as an attacker requires stego keys to extract hidden information. The proposed method is applied on images of different formats: BMP, JPG and PNG. Experimental results show that proposed method gives better security and better image quality.

Dr. Manjot Kaur Bhatia is working as a Professor in Department of Information technology at Jagan Institute of Management Studies, Delhi, India. She is MPhil, MCA and PhD (Computer Science) from University of Delhi. She has more than fourteen years of teaching experience in the areas of Operating system, Databases, Grid computing, Linux etc. Her research areas include Grid Computing and Information Security on which papers have been published in various National and International conferences and journals.