Tạp chí Tin học và Điều khiển học, T.21, S.4 (2005), 281–292

VỀ MỘT CẢI TIẾN ĐỐI VỚI LƯỢC ĐỒ GIẤU DỮ LIỆU AN TOÀN VÀ VÔ HÌNH TRONG CÁC BỨC ẢNH HAI MÀU

BÙI THẾ HỒNG

Viện Công nghệ thông tin

Abstract. In [1], Yu-Chee Tseng and Hsiang-Kuang Pan proposed a steganography scheme for hiding critical information in a host binary images. This scheme is revised from the scheme in [2] to maintain higher quality of the host image by sacrificing some data hiding space but also ensures that in each $m \times n$ image block of the host image, as many as $\lfloor log_2(mn+1) \rfloor - 1$ bits can be hidden in the block by changing at most 2 bits in the block. In the scheme, a distance matrix $dist(F)$ is defined as a condition to decide whether a bit of image block F_i of the host image F can be complemented for hiding data. This paper proposes a revised scheme that replaces the distance matrix by a new one called complement able bit matrix. The new scheme reduces the run time of the hiding algorithm and maintains the host image quality.

Tóm tắt. Trong bài báo [1], Yu-Chee Tseng và Hsiang-Kuang Pan đã đưa ra một lược đồ giấu tin dùng để giấu các thông tin quan trọng vào một ảnh chủ nhị phân. Lược đồ này được cải tiến từ lược đồ trong [2] nhằm giữ được chất lượng của ảnh chủ bằng cách chịu mất đi một không gian giấu dữ liệu nào đó nhưng vẫn đảm bảo trong mỗi khối ảnh chủ cỡ $m \times n$ có thể giấu được đến $\lfloor log_2(mn+1) \rfloor - 1$ bit mà chỉ cần thay đổi nhiều nhất là 2 bit. Điểm mới của lược đồ này là sử dụng một ma trận khoảng cách làm tiêu chí trong việc lựa chọn bit để giấu dữ liệu. Bài báo này trình bày một lược đồ giấu tin được cải tiến từ lược đồ nói trên bằng cách thay ma trận khoảng cách bằng một ma trận khác được gọi là ma trận bit có thể đảo. Lược đồ mới này sẽ giảm đáng kể thời gian chạy của thuật toán giấu tin và luôn giữ được chất lượng của ảnh chủ.

I. GIỚI THIỆU

Giấu dữ liệu là một trong các kỹ thuật truyền tin an toàn trong kỷ nguyên thông tin số. Người ta giấu một thông điệp quan trọng vào một phương tiện chủ bằng cách thay đổi một số bit thông tin nào đó của phương tiện này mà không làm ảnh hưởng nhiều lắm đến chất lượng của nó. Với một ảnh màu, việc giấu tin có thể được thực hiện bằng một cách tiếp cận rất đơn giản là sử dụng các bit ít ý nghĩa nhất (LSB) của mỗi điểm ảnh để giấu. Ảnh sử dụng càng nhiều bit cho một điểm càng có khả năng giấu được nhiều thông tin và càng khó bị phát hiện.

Bài toán giấu tin trong ảnh hai màu (đen trắng) phức tạp hơn nhiều bởi vì nếu không khéo thì chỉ cần thay đổi một bit người ta cũng có thể dễ dàng nhận ra. Giải pháp duy nhất cho vấn đề này là chia bức ảnh thành các khối có kích thước thích hợp để có thể giấu một hoặc một vài bit. Kích thước của khối không được nhỏ quá để dễ bị phát hiện và cũng không

được lớn quá để chỉ giấu được rất ít.

Với một khối ảnh kích thước $m \times n$, lược đồ trong [2] giấu được tối đa $\log_2(mn+1)$ bit mà chỉ cần thay đổi nhiều nhất là 2 bit trong khối. Tuy nhiên, lược đồ này có một nhược điểm là không đảm bảo được chất lượng của ảnh chủ, nhất là đối với các khối toàn đen hoặc toàn trắng.

Lược đồ đề xuất trong [1] đã khắc phục được nhược điểm này bằng cách đưa thêm một đại lượng nữa, gọi là khoảng cách $[dist(F)]_{i,j}$ mà giá trị của nó được sử dụng để làm điều kiện quyết định liệu một khối ảnh chủ có thể dùng để giấu dữ liệu hay không. Tuy nhiên, theo chúng tôi thấy, để tính $[dist(F)]_{i,j}$ không cần phải sử dụng tất cả các điểm của F mà chỉ cần các điểm lân cận của i, j là đủ.

Bài báo này trình bày một lược đồ giấu tin có cải tiến cách tính khoảng cách từ $[F]_{i,j}$ đến phần tử $[F]_{x,y}$ gần nhất có giá trị khác với giá trị của $[F]_{i,j}$. Lược đồ mới sẽ giảm đáng kể thời gian chạy của thuật toán.

Các phần tiếp theo sẽ trình bày các ý chính của thuật toán trong [1] và một số cải tiến nhằm nâng cao hiệu năng của thuật toán này.

2. LƯỢC ĐỒ TSENG-PAN VÀ MỘT SỐ NHẬN XÉT

Bài toán giấu dữ liệu trong ảnh có thể được định nghĩa như sau:

Cho một ảnh chủ H và một thông điệp khách G, một lược đồ giấu dữ liệu cần có một hàm giấu dữ liệu S_h và một hàm tìm lại dữ liệu S_r sao cho

$$H' = S_h(H, G, K),$$

$$S_r(H', K) = S_r(S_h(H, G, K), K) = G,$$

trong đó, K là một khóa bí mật. Tức là, S_r có thể gỡ thông điệp khách đã được giấu ra khỏi ảnh chủ. Hơn nữa, để đánh lừa đối phương, cần phải che giấu làm sao cho H' không được khác với H quá nhiều.

2.1. Lược đồ Tseng-Pan

Vì chúng ta làm việc trên các ảnh 2 màu, nên việc thay bất kỳ một bit nào trong ảnh cũng có thể dễ dàng bị phát hiện ra. Do đó, để đảm bảo chất lượng của ảnh, chắc chắn chúng ta nên càng thay đổi ít bit càng tốt. Một khối ảnh chủ hoàn toàn đen hoặc hoàn toàn trắng sẽ không được sử dụng để giấu tin. Ngoài ra, nếu một bit phải bị thay đổi, chúng ta hy vọng là vị trí của nó rất gần với một bit khác có giá trị bằng giá trị mới của bit này. Chẳng hạn, chúng ta xét một ảnh F được biểu diễn bằng một ma trận bit cỡ 5×5, F được sửa đổi thành hai ảnh F' và F'' như sau:

$F =$

1	1	0	0	0
1	1	0	0	0
1	0	0	0	0
1	0	0	0	0
1	0	0	0	0

$F' =$

1	1	0	0	0
1	1	0	0	0
1	**1**	0	0	0
1	0	0	0	0
1	0	0	0	0

$F'' =$

1	1	0	0	0
1	1	0	0	0
1	0	0	0	0
1	0	0	**1**	0
1	0	0	0	0

Cả F' và F'' đều chỉ khác với F ở một bit. Chúng ta thấy, F và F' giống nhau hơn so với F và F'', bởi vì F'' khác hẳn với F tại điểm $(4,4)$. Việc thay bit này bằng 1 rất lộ vì lân cận với nó toàn là 0.

Để hình thức hóa quan sát trên, các tác giả bài báo [1] đã đưa ra một ma trận gọi là $dist(F)$ có cùng cỡ với F với các phần tử được xác định như sau:

$$[dist(F)]_{i,j} = \min_{\forall x,y}\{\sqrt{|i-x|^2+|j-y|^2}|[F]_{i,j} \neq [F]_{x,y}\},$$

nghĩa là, $[dist(F)]_{i,j}$ là khoảng cách từ $[F]_{i,j}$ tới phần tử $[F]_{x,y}$ gần nhất có giá trị khác với $[F]_{i,j}$. Sau này, ma trận khoảng cách sẽ được sử dụng làm một tiêu chí để chọn bit bị thay đổi. Theo công thức định nghĩa ở trên, ta có thể tính được ma trận khoảng cách của F trong ví dụ trên như sau:

$$dist(F) = \begin{bmatrix} 2 & 1 & 1 & 2 & 3 \\ \sqrt{2} & 1 & 1 & 2 & 3 \\ 1 & 1 & \sqrt{2} & \sqrt{5} & \sqrt{10} \\ 1 & 1 & 2 & \sqrt{8} & \sqrt{13} \\ 1 & 1 & 2 & 3 & 4 \end{bmatrix}$$

Lược đồ trong [1] nhằm mục đích giấu một dãy bit vào một bức ảnh chủ nhị phân F. Ảnh F sẽ được phân hoạch thành các khối ảnh nhỏ F_i có cỡ $m \times n$. Để cho đơn giản, có thể giả thiết rằng cỡ của F là bội của $m \times n$. Trong mỗi khối F_i, chúng ta sẽ thử giấu r bit dữ liệu, với $r \leqslant [\log_2(mn+1)] - 1$.

Gọi $b_1b_2...b_r$ là chuỗi bit cần phải giấu trong khối F_i, và F_i' là khối ảnh sau khi giấu.

Một khóa bí mật bao gồm hai thành phần:

K: một ma trận nhị phân cỡ $m \times n$ được chọn ngẫu nhiên.

W: một ma trận trọng số cỡ $m \times n$ với

$$\{[W]_{j,k},\ j = 1,...,m, k = 1,...,n\} = \{1,2,3,...,2^{r+1}-1\},$$

và mọi khối con cỡ 2×2 của W đều chứa ít nhất một phần tử lẻ.

Một ma trận trọng số được định nghĩa như trên là hoàn toàn có thể chọn được. Tính chất thứ nhất khẳng định các phần tử của ma trận trọng số chỉ là các số tự nhiên nhỏ hơn hoặc bằng $2^{r+1}-1$. Để đảm bảo tính chất thứ hai, chúng ta có thể có một số cách chọn, chẳng hạn chọn xen kẽ một giá trị chẵn rồi một giá trị lẻ hoặc chọn một hàng chẵn rồi một hàng lẻ. Hai tính chất này của ma trận trọng số là các điều kiện giúp chúng ta có thể tìm lại các thông tin đã giấu trong một khối. Ví dụ sau đây chỉ ra ba cách để lập ra một ma trận trọng số cỡ 4×4:

$$\begin{bmatrix} o & e & o & e \\ e & o & e & o \\ o & e & o & e \\ e & o & e & o \end{bmatrix} \quad \begin{bmatrix} o & o & o & o \\ e & e & e & e \\ o & o & o & o \\ e & e & e & e \end{bmatrix} \quad \begin{bmatrix} e & o & e & o \\ o & e & o & e \\ e & o & e & o \\ o & e & o & e \end{bmatrix}$$

trong đó, "o" là ký hiệu một số lẻ, "e" ký hiệu một số chẵn. Số các ma trận trong số có thể được lập ra theo một trong ba cách trên là:

$$\lfloor C_{mn/2}^{2^{r-1}} \times (2^{r-1})! \times (2^{r-1})^{(mn/2)-(2^{r-1})} \rfloor \times \lfloor C_{mn/2}^{2^{r-1}} \times (2^{r-1}-1)! \times (2^{r-1}-1)^{(mn/2)-(2^{r-1}+1)} \rfloor$$

trong đó, phần thứ nhất là các khả năng dùng để chọn các số lẻ, còn phần thứ hai là các khả năng dùng để chọn các số chẵn.

Các bước của quá trình giấu dữ liệu của lược đồ này được thực hiện như sau:
B1. Nếu khối ảnh F_i hoàn toàn đen hoặc hoàn toàn trắng thì giữ nguyên khối này (không sử dụng để giấu dữ liệu) và chuyển sang xét khối tiếp theo. Ngược lại, thực hiện các bước sau.
B2. Tính tổng

$$SUM((F_i \oplus K) \otimes W) \tag{1}$$

trong đó các phép toán $\oplus, \otimes$ và SUM được định nghĩa như sau:

- $\oplus$ là phép OR loại trừ từng cặp bit của hai ma trận nhị phân cùng cỡ. Ví dụ,

$$F_i = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}; \quad K = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}; \quad W = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix}$$

$$\Rightarrow F_i \oplus K = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \oplus \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

- $\otimes$ là phép nhân theo cặp phần tử của hai ma trận cùng cỡ. Ví dụ,

$$(F_i \oplus K) \otimes W = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 2 & 0 \\ 1 & 0 & 3 \end{bmatrix}$$

- SUM là tổng số học của tất cả các phần tử của một ma trận. Ví dụ,

$$SUM((F_i \oplus K) \otimes W) = (1+0+3) + (0+2+0) + (1+0+3) = 10$$

B3. Từ ma trận $F_i \oplus K$, với mỗi $w = 1, 2, ..., 2^{r+1} - 1$ tính tập hợp sau:

$$\begin{aligned} S_w = \big\{(j,k) | ([W]_{j,k} = w \wedge (F_i \oplus K)_{j,k} = 0) \wedge ([dist(F)]_{j',k'} \leqslant \sqrt{2}) \\ \vee ([W]_{j,k} = 2^{r+1} - w \wedge (F_i \oplus K)_{j,k} = 1) \wedge ([dist(F)]_{j',k'} \leqslant \sqrt{2}) \big\} \end{aligned}$$

trong đó, $[dist(F)]_{j',k'}$ là phần tử của ma trận $dist(F)$ tương ứng với bit $[F_i]_{j,k}$ trong khối F_i còn $\wedge$ và $\vee$ tương ứng là các phép AND và OR logic thông thường. Ngoài ra, chúng ta cũng quy định là $S_w = S'_w$ với mọi $w \equiv w' (\mathrm{mod} 2^{r+1})$.

Có thể thấy ngay rằng với mỗi w như trên, S_w là một tập chỉ chứa các chỉ số (j,k) sao cho nếu đảo bit $[F_i]_{j,k}$ thì tổng (1) sẽ tăng thêm một lượng bằng w. Quả vậy, nếu mệnh đề thứ nhất trong định nghĩa của S_w thỏa mãn, tức là $[W]_{j,k} = w$ và $(F_i \oplus K)_{j,k} = 0$, thì khi đảo $[F_i]_{j,k}$, tổng (1) sẽ được tăng thêm một lượng bằng w. Ngược lại, nếu $[W]_{j,k} = 2^{r+1} - w$ và $(F_i \oplus K)_{j,k} = 1$ thì khi đảo $[F_i]_{j,k}$, tổng (1) sẽ bị giảm đi một lượng bằng $2^{r+1} - w$, tương đương với tăng thêm một lượng bằng w khi môđun cho 2^{r+1}.

Các bổ đề sau đây tổng hợp một số tính chất quan trọng của những tập này:

Bổ đề 1. *Với mọi $w = 1, 2, ..., 2^{r+1} - 1$ và $w \neq 2^r$ ta luôn có mệnh đề $S_w = \emptyset \Rightarrow (S_{2^{r+1}-w} \neq \emptyset)$.*

Chứng minh. Với $w \neq 2^r$, giả sử $S_w = \emptyset$. Từ định nghĩa của ma trận trọng số ta suy ra phải tồn tại ít nhất một phần tử $[W]_{j,k} = w$ sao cho $[F_i]_{j,k} \wedge [K]_{j,k} = 1$ vì nếu không tức là $[F_i]_{j,k}$

phải bằng 0 và khi đảo nó sẽ thành 1, dẫn đến tổng (1) sẽ tăng lên w và do đó S_w sẽ không rỗng (trái với giả thiết). Bởi vậy, nếu ta đảo giá trị của $[F_i]_{j,k}$ thì tổng (1) sẽ giảm đi w hay tăng lên $2^{r+1} - w(\text{mod}2^{r+1})$, suy ra $S_{2^{r+1}} = \emptyset$. ■

Bổ đề 2. *Tập hợp* $S_{2^r} \neq \emptyset$.

Chứng minh. Từ định nghĩa của W, ta suy ra tồn tại ít nhất một phần tử $[W]_{j,k} = 2^r$. Mặt khác, vì $2^r \equiv -2^r(\text{mod}2^{r+1})$ nên nếu ta đảo giá trị của $[F_i]_{j,k}$ thì tổng (1) sẽ tăng lên hoặc giảm đi 2^r, có nghĩa là $S_{2^r} \neq \emptyset$. ■

Trong định nghĩa của S_w, việc kiểm soát chất lượng của ảnh đã được đưa vào nhờ ma trận khoảng cách. Một bit chỉ có thể được đảo nếu tồn tại ít nhất một bit lân cận khác với nó. Ở đây, khoảng cách tối đa giữa một bit và các lân cận của nó được xác định là $\sqrt{2}$, tức một bit có 8 bit lân cận.

B4. Xác định hiệu trọng số

$$d \equiv (b_1 b_2 ... b_r 0) - SUM((F_i \oplus K) \otimes W)(\text{mod}2^{r+1}).$$

Nếu $d = 0$, không cần phải thay đổi F_i mà coi như đã giấu được chuỗi bit vào khối này. Ngược lại, chúng ta chạy chương trình sau để chuyển F_i thành F_i'.

if (tồn tại một số $h \in \{0, 1, ..., 2^{r+1} - 1\}$ sao cho $S_{hd} \neq \emptyset$ và $S_{-(h-1)d} \neq \emptyset$)
then
    Chọn ngẫu nhiên một số h để điều kiện trên thỏa mãn;
    Chọn ngẫu nhiên một cặp $(j, k) \in S_{hd}$ và đảo bit $[F_i]_{j,k}$; *tăng tổng (1) thêm hd*
    Chọn ngẫu nhiên một cặp $(j, k) \in S_{-(h-1)d}$ và đảo bit $[F_i]_{j,k}$; *giảm (1) đi $(hd - d)$*
else *không dữ liệu nào được giấu*
    if $(SUM((F_i \oplus K) \otimes W)(\text{mod}2) \equiv 1)$ then
        Giữ F_i không đổi
    else
        Chọn (j, k) sao cho $[W]_{j,k}$ là lẻ và $[dist(F)]_{j',k'}$ tương ứng của nó là nhỏ nhất.
        Đảo bit $[F_i]_{j,k}$ /* tăng tổng (1) lên một số lẻ *
    endif;
endif;

Để đơn giản trong khi trình bày thuật toán trên, chúng tôi đã bỏ qua một điểm cần phải lý giải. Đó là, có thể sẽ có các số h và d, ví dụ như $h = 0, h = 1$ hoặc $d = 0$, làm xuất hiện tập hợp chưa được định nghĩa S_0. Thực ra cũng giống như các tập S_w khác, chúng ta có thể coi S_0 là tập chứa chỉ số của những bit trong F_i mà nếu đảo chúng thì sẽ làm tổng (1) tăng lên 0 đơn vị. Vì có thể đạt được điều đó bằng cách không làm gì trên F_i, nên chúng ta có thể luôn luôn coi S_0 là một tập không rỗng và trong trường hợp này, bất cứ khi nào gặp câu lệnh "đảo bit $[F_i]''_{j,k}$ ta có thể bỏ qua để sang lệnh tiếp theo.

Đến đây chúng ta thấy tuy gọi là "giấu dữ liệu" nhưng thực ra chúng ta không giấu gì cả mà chỉ tìm cách thay đổi F_i để đảm bảo cho điều kiện sau được thỏa mãn thì có nghĩa là dữ liệu được "giấu":

I1:

$$SUM((F_i' \oplus K) \otimes W) \equiv b_1 b_2 ... b_r 0(\text{mod}2^{r+1}).$$

Quả vậy, nếu may mắn mà $d = 0$ thì không cần phải thay đổi gì ở F_i ta đã có ngay được I1 và coi như *đã giấu được* chuỗi bit vào F_i. Ngược lại, nếu $d \neq 0$ thì phải tìm cách thay đổi F_i để tăng tổng (1) thêm d thì hiệu trọng số mới sẽ bằng 0 và I1 sẽ thỏa mãn. Trong trường hợp này, có thể xảy ra một trong hai khả năng phụ thuộc vào việc liệu có tồn tại một số h như đã xác định trong mệnh đề if ngoài hay không. Nếu có một số h như vậy thì đảo bit thứ nhất sẽ làm tổng (1) tăng thêm hd và đảo bit thứ hai sẽ làm tổng (1) tăng thêm $-(h-1)d$. Kết quả cuối cùng tổng (1) sẽ tăng thêm d. Từ đó suy ra I1 thỏa mãn và xem như đã giấu được chuỗi bit vào F_i. Trong trường hợp ngược lại (không tìm được h) thì xem như không giấu được chuỗi bit. Đến đây có thể xảy ra khả năng tuy không tìm được h nhưng I1 lại đúng. Khi ấy, cần phải làm cho I1 trở thành sai để phân biệt với các trường hợp giấu được. Ta thấy là nếu I1 đúng thì tổng $SUM((F_i' \oplus K) \otimes W)$ phải chẵn vì chuỗi $b_1b_2...b_r0$ biểu diễn một số chẵn. Do đó, để I1 trở thành sai thì chỉ cần làm cho tổng này trở thành lẻ. Mệnh đề if trong của bước B4 nhằm thực hiện công việc này. Vấn đề còn lại là phải chỉ ra một cách thức để tìm được một số h như đã xác định trong bước này. Chúng ta sẽ chứng minh điều đó trong bổ đề sau:

Bổ đề 3. *Bước B4 luôn luôn thực hiện được và để giấu được chuỗi bit chỉ cần đảo nhiều nhất là 2 bit của F_i.*

Chứng minh. Chúng ta sẽ thử với các số h khác nhau và chỉ ra rằng có thể tìm được một số h thỏa mãn điều kiện mong muốn trong mệnh đề *if ngoài*. Đầu tiên, chúng ta sẽ kiểm tra S_d. Nếu $S_d \neq \emptyset$ thì $h = 1$ sẽ thỏa mãn điều kiện mong muốn vì $S_{-(h-1)d} = S_0 \neq \emptyset$. Ngược lại, $S_d = \emptyset$ sẽ kéo theo $S_{-d} \neq \emptyset$ (Bổ đề 1). Xét tiếp S_{2d}. Nếu $S_{2d} \neq \emptyset$ thì $h = 2$ là số cần tìm vì $S_{-(h-1)d} = S_{-d} \neq \emptyset$. Ngược lại, $S_{2d} = \emptyset$ sẽ kéo theo $S_{-2d} \neq \emptyset$ (Bổ đề 1). Xét tiếp S_{3d}. Nếu $S_{3d} \neq \emptyset$ thì $h = 3$ là số cần tìm vì $S_{-(h-1)d} = S_{-2d} \neq \emptyset$. Tiếp tục làm như vậy với $S_{4d}, S_{5d}, ...$ và nếu vẫn không tìm thấy thì cuối cùng sẽ xét đến S_{2^r}. Theo Bổ đề 2 thì tập này luôn luôn khác rỗng và vì vậy ta sẽ tìm được số h thỏa mãn điều kiện mong muốn. Bổ đề 3 được chứng minh. ■

Cần lưu ý, nếu kết quả F_i' là hoàn toàn đen hoặc hoàn toàn trắng thì việc giấu dữ liệu được coi là không thành và chúng ta sẽ thử giấu chuỗi bit này vào khối tiếp theo.

Từ I1 và Bổ đề 3, chúng ta có thể rút ra hai bất biến sau đây. Sau khi thực hiện Bước 4, nếu khối F_i' không hoàn toàn đen hoặc không hoàn toàn trắng thì:

I2: Nếu $SUM((F_i' \oplus K) \otimes W)$ chẵn thì $SUM((F_i' \oplus K) \otimes W)/2 \equiv b_1b_2...b_r (\mathrm{mod} 2^{r+1})$.

I3: Nếu $SUM((F_i' \oplus K) \otimes W)$ lẻ thì không có chuỗi bit nào được giấu trong F_i'.

S5. Khi nhận được khối F_i', người nhận sẽ tính được dữ liệu giấu trong khối là $SUM((F_i' \oplus K) \otimes W)/2$ nếu F_i' không hoàn toàn đen hoặc không hoàn toàn trắng và $SUM((F_i' \oplus K) \otimes W)$ là chẵn. Ngược lại, F_i' không chứa dữ liệu giấu.

Trong lược đồ này có hai tính chất quan trọng luôn được bảo toàn. Thứ nhất, nó đảm bảo rằng một bit chỉ có thể được đảo khi có ít nhất một bit lân cận có giá trị bằng giá trị mới của nó. Điều này đảm bảo cho việc thay đổi bit hoàn toàn không bị nhận ra. Chúng ta có thể chứng minh tính chất này qua việc xem xét câu lệnh if trong bước B4. Nếu điều kiện trong if là đúng thì định nghĩa của S_w trong bước B3 đã đảm bảo cho ta tính chất này. Ngược lại, trong mệnh đề else sẽ xảy ra hai trường hợp, hoặc F_i sẽ được giữ nguyên hoặc

một bit của F_i được chọn ngẫu nhiên sao cho phần tử tương ứng với nó trong W là lẻ và trong $dist[F]$ là nhỏ nhất sẽ được đảo. Vì F_i không hoàn toàn đen hoặc không hoàn toàn trắng nên trong F_i phải tồn tại ít nhất hai bit lân cận nhau có giá trị đối nhau. Do đó, đối với một khối con cỡ 2×2 bất kỳ có chứa hai bit này, giá trị (khoảng cách) của các phần tử của khối con 2×2 tương ứng trong $dist(F)$ đều không lớn hơn $\sqrt{2}$. Từ định nghĩa của ma trận trọng số, khối con 2×2 trong W tương ứng với khối con trong F_i phải chứa một phần tử có giá trị lẻ. Do đó, bit cần đảo phải có một bit lân cận có giá trị bằng giá trị mới của nó.

Thứ hai, để đảm bảo chất lượng ảnh sau khi thay đổi, chúng ta đã không sử dụng các khối hoàn toàn đen hoặc hoàn toàn trắng để giấu dữ liệu. Tuy nhiên, trong bước B4, một khối sau khi bị thay đổi có thể trở thành toàn đen hoặc toàn trắng. Trong trường hợp này, để tránh nhầm lẫn với các khối gốc toàn đen hoặc toàn trắng, chúng ta vẫn giữ nguyên như vậy và giấu lại các thông tin đã giấu trong khối này vào khối gốc tiếp theo.

2.2. Một số nhận xét về lược đồ Tseng-Pan

- Trong lược đồ trên đây, các tác giả đã đưa ra một tiêu chuẩn để kiểm soát chất lượng của ảnh chủ sau khi giấu. Tiêu chuẩn này là: một bit chỉ có thể được đảo nếu có ít nhất một bit lân cận có giá trị khác với nó. Tức là, sau khi đảo thì bit này hòa lẫn được với các bit xung quanh nhằm đánh lừa sự chú ý của người khác. Với cách làm như vậy, các khối ảnh gốc toàn đen hoặc toàn trắng sẽ không được sử dụng để giấu tin và do vậy làm giảm dung lượng có thể giấu nhưng bù lại nó sẽ làm tăng chất lượng của ảnh sau khi giấu so với lược đồ trước đó.

- Ngoài ra, sau khi biến đổi mà khối ảnh gốc lại trở thành toàn đen hoặc toàn trắng thì khối này cũng bị coi là không giấu thông tin (cho dù đã giấu) để tránh nhầm lẫn với các khối gốc toàn đen hoặc toàn trắng đã bị loại ra ngay từ bước B1.

- Khi định nghĩa ma trận $dist(F)$ các tác giả đã đưa ra một khái niệm được gọi là khoảng cách tại một bit. Khoảng cách tại điểm (i, j) là bằng khoảng cách ngắn nhất từ điểm này đến tất cả các điểm có giá trị khác với giá trị của nó trong ma trận bit F. Nhưng khi vận dụng vào bước B3 trong lược đồ thì lại chỉ sử dụng có 8 điểm lân cận của điểm trong F tương ứng với điểm đang xét trong ma trận F_i là đủ. Từ đây nảy ra một ý tưởng liệu có thể thay điều kiện khoảng cách của bit tương ứng trong ma trận F nhỏ hơn hoặc bằng $\sqrt{2}$ bằng việc xét xem trong lận cận của bit này có bit nào có giá trị khác với giá trị của nó hay không. Nếu làm được như vậy thì đã giảm được đáng kể thời gian tính ma trân $dist(F)$.

3. LƯỢC ĐỒ TSENG-PAN CẢI TIẾN

3.1. Kiểm soát chất lượng ảnh sau khi giấu tin

Trước hết, chúng tôi đưa ra một khái niệm mới thay thế khái niệm khoảng cách trong lược đồ Tseng-Pan.

Như đã thấy trong lược đồ Tseng-Pan, để đảm bảo chất lượng của ảnh chủ, các khối ảnh hoàn toàn đen hoặc hoàn toàn trắng sẽ không được sử dụng để giấu tin. Ngoài ra, nếu một bit phải bị thay đổi, chúng ta hy vọng là vị trí của nó rất gần với một bit khác có giá trị bằng giá trị mới của bit này. Để thể hiện điều đó, chúng tôi đưa ra một ma trận bit có cỡ

bằng ma trận bit của ảnh F được gọi là $comable(F)$ (có thể đảo - complement able), trong đó một bit của ma trận này sẽ được đặt bằng 0 nếu bit tương ứng với nó trong F không có một bit lân cận nào có giá trị khác với bit này, ngược lại bit này của ma trận có thể đảo sẽ được đặt bằng 1. Có thể hình thức hóa ý tưởng trên bằng định nghĩa sau.

Cho một ma trận ảnh F cỡ $M \times N$, ma trận bit có thể đảo của F là một ma trận bit có cỡ bằng cỡ của ma trận F với các phần tử được xác định như sau:

$$[comable(F)]_{i,j} = 0 \text{ nếu } [F]_{i,j} = [F]_{i+x,j+y}, \text{ với mọi } x, y \in \{-1, 0, 1\}$$
$$\text{và } 0 < i + x \leqslant M,\ 0 < j + y \leqslant N,$$
$$[comable(F)]_{i,j} = 1 \text{ nếu ngược lại.}$$

Nghĩa là, $[comable(F)]_{i,j}$ sẽ cho ta biết liệu bit tương ứng của F có thể đảo mà không làm ảnh hưởng đến chất lượng của ảnh F hay không. Nếu $[comable(F)]_{i,j} = 1$ thì bit (i, j) của F có thể đảo, còn ngược lại nếu đảo bit này thì sẽ làm ảnh hưởng lớn đến chất lượng của F. Ma trận này sẽ được sử dụng để làm tiêu chuẩn chọn lựa các bit sẽ được đảo để giấu dữ liệu.

Theo định nghĩa trên, chúng ta có thể tính ma trận bit có thể đảo của ma trận ảnh F đã cho trong ví dụ ở Mục 1.

$$F = \begin{vmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{vmatrix};\ dist(F) = \begin{vmatrix} 2 & 1 & 1 & 2 & 3 \\ \sqrt{2} & 1 & 1 & 2 & 3 \\ 1 & 1 & \sqrt{2} & \sqrt{5} & \sqrt{10} \\ 1 & 1 & 2 & \sqrt{8} & \sqrt{13} \\ 1 & 1 & 2 & 3 & 4 \end{vmatrix};\ Comable(F) = \begin{vmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{vmatrix}$$

Chúng ta thấy ở các vị trí mà ma trận $comable(F)$ có giá trị 1 thì ở các vị trí tương ứng trong ma trận $dist(F)$ có giá trị nhỏ hơn hoặc bằng $\sqrt{2}$. Ngược lại, nếu ở các vị trí mà $comable(F)$ có giá trị 0 thì ở các vị trí tương ứng trong ma trận $dist(F)$ có giá trị lớn hơn $\sqrt{2}$. Do đó, chúng ta có thể sử dụng $comable(F)$ thay cho $dist(F)$ trong lược đồ giấu tin.

Việc tính các phần tử của ma trận bit có thể đảo đơn giản và nhanh hơn rất nhiều so với việc tính ma trận khoảng cách trước đây, nhất là khi cỡ của F rất lớn. Để tính $[dist(F)]_{i,j}$, trước hết phải tính khoảng cách từ bit này tới tất cả các bit của F có giá trị khác với nó và sau đó chọn khoảng cách nhỏ nhất. Thời gian tính $dist(F)$ có bậc $(MN)^2$. Trong khi đó, để tính $[comable(F)]_{i,j}$, ta chỉ cần tối đa 8 phép so sánh, do đó thời gian để tính $comable(F)$ chỉ có bậc MN.

Để tính giá trị của $[comable(F)]_{i,j}$, ta có thể sử dụng hàm sau:

```
function comable(i, j)
  for x ← −1 to 1 do
    if i + x > 0 and i + x ⩽ m then
      for y ← −1 to 1 do
        if j + y > 0 and j + y ⩽ n and F[i, j] ≠ F[i + x, j + y] then return 1
  return 0
```

3.2. Lược đồ Tseng-Pan cải tiến

Sử dụng tính chất của ma trận bit có thể đảo, chúng ta xây dựng một lược đồ giấu tin được cải tiến từ lược đồ Tseng-Pan như sau:

B1. Nếu F_i hoàn toàn đen hoặc hoàn toàn trắng thì giữ nguyên khối này và không sử dụng nó để giấu dữ liệu. Chuyển sang xét khối tiếp theo. Ngược lại, thực hiện các bước sau.

B2. Tính $SUM((F_i \oplus K) \otimes W)$.

B3. Từ ma trận $F_i \oplus K$, với mỗi $w = 1, 2, ..., 2^{r+1} - 1$, tính tập hợp sau:

$$S_w = \{(j,k) | ([W]_{j,k} = w \wedge (F_i \oplus K)_{j,k} = 0) \wedge ([Comable(F)]_{j',k'} = 1)$$
$$\vee ([W]_{j,k} = 2^{r+1} - w \wedge (F_i \oplus K)_{j,k} = 1) \wedge ([Comable(F)]_{j',k'} = 1)\}$$

trong đó, $[Comable(F)]_{j',k'}$ là phần tử của ma trận $Comable(F)$ tương ứng với bit (j,k) trong khối F_i. Ngoài ra, chúng ta cũng qui định là $S_w = S'_w$ với mọi $w \equiv w' (\text{mod} 2^{r+1})$. Như vậy, S_w chỉ chứa các toạ độ (j,k) mà tại vị trí tương ứng với vị trí này trong ma trận bit có thể đảo có giá trị bằng 1. Tức là bit này trong ảnh chủ F có ít nhất một bit lân cận có giá trị khác với nó. Khi cần đảo bit để giấu dữ liệu, chúng ta sẽ chỉ chọn các bit có tọa độ nằm trong S_w để đảm bảo chất lượng của ảnh chủ.

B4. Xác định hiệu trọng số

$$d \equiv (b_1 b_2 ... b_r 0) - SUM((F_i \oplus K) \otimes W) (\text{mod} 2^{r+1}).$$

Nếu $d = 0$, không cần phải thay đổi F_i mà coi như đã giấu được chuỗi bit vào khối này.
Ngược lại, chúng ta chạy chương trình sau để chuyển F_i thành F'_i.
if (tồn tại một số $h \in \{0, 1, ..., 2^{r+1} - 1\}$ sao cho $S_{hd} \neq \emptyset$ và $S_{-(h-1)d} \neq \emptyset$)
then
    Chọn ngẫu nhiên một số h để điều kiện trên thỏa mãn;
    Chọn ngẫu nhiên một cặp $(j,k) \in S_{hd}$ và đảo bit $[F_i]_{j,k}$;
    Chọn ngẫu nhiên một cặp $(j,k) \in S_{-(h-1)d}$ và đảo bit $[F_i]_{j,k}$;
else /* không dữ liệu nào được giấu */
    if $(SUM((F_i \oplus K) \otimes W) (\text{mod} 2) \equiv 1)$ then
        Giữ F_i không đổi.
    else
        Chọn (j,k) sao cho $[W]_{j,k}$ là lẻ và $[Comable(F)]_{j',k'}$ tương ứng với nó bằng 1.
        Đảo bit $[F_i]_{j,k}$
    endif;
endif;

Nếu kết quả F'_i là hoàn toàn đen hoặc hoàn toàn trắng thì việc giấu dữ liệu được coi là không thành và chúng ta sẽ thử giấu lại chuỗi bit này vào khối tiếp theo.

B5. Khi có được khối F'_i, người nhận tính dữ liệu giấu trong khối này là $SUM((F'_i \oplus K) \otimes W)/2$ nếu F'_i không hoàn toàn đen hoặc không hoàn toàn trắng và $SUM((F'_i \oplus K) \otimes W)$ là chẵn. Ngược lại, F'_i không chứa dữ liệu giấu.

3.3. Ví dụ minh họa

Cho ma trận ảnh nhị phân F cỡ 8×8 và các ma trận K và W:

$F =$

	F_1			F_2			
0	1	0	1	0	1	1	0
1	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0
1	1	1	0	1	0	0	0
0	0	1	1	1	0	1	0
1	1	0	1	0	1	1	1
1	0	1	1	0	1	1	1
	F_3			F_4			

$K =$

1	1	0	0
0	1	0	0
1	1	1	0
0	0	1	0

$W =$

1	3	5	7
2	4	6	8
9	11	13	15
10	12	14	2

Chọn $m = n = 4$. Vì $\lfloor log_2(mn + 1) \rfloor = 4$, nên có thể chọn $r = 3$. Ma trận khóa K và ma trận trọng số W được chọn như trong hình trên. Ma trận W đảm bảo hai tính chất: mọi phần tử của nó đều lớn hơn hoặc bằng 1 và nhỏ hơn hoặc bằng $2^{r+1} - 1 = 15$, trong mỗi khối 2×2 đều có ít nhất một số lẻ.

Vì $r = 3$ nên có thể giấu trong mỗi khối 3 bit. Như vậy, có thể giấu tối đa là 12 bit vào F. Giả sử dãy bit cần giấu là $B = 110111011001$. Ta sẽ thử dùng lược đồ cải tiến để giấu lần lượt 3 bit của B vào các khối ảnh con F_i.

Bước 1 và Bước 2: Tính

$Comable(F)$

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	0

	$F_1 \oplus K$			$F_2 \oplus K$			
1	0	0	1	1	0	1	0
1	1	0	0	1	0	1	1
0	1	1	0	1	1	1	0
0	0	1	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	1	1	1	1	1	0
0	0	1	1	1	0	0	1
1	0	0	1	0	1	0	1
	$F_3 \oplus K$			$F_4 \oplus K$			

$SUM((F_1 \oplus K) \otimes W) = 52$ $SUM((F_2 \oplus K) \otimes W) = 55$

$SUM((F_3 \oplus K) \otimes W) = 63$ $SUM((F_4 \oplus K) \otimes W) = 53$

Bước 3 và Bước 4:

+ Đối với F_1 : $SUM((F_1 \oplus K) \otimes W)(\text{mod}2^4) = 52(\text{mod}16) = 4$

Vì chuỗi 3 bit cần giấu đầu tiên là 110, nên ta có: $d = (1100) - 4 = 12 - 4 = 8$.

Vì vậy, sẽ phải thay đổi các bit của F_1 để tăng trọng số lên 8.

Vì $[W]_{2,4} = 8$ và $[F_1 \oplus K]_{2,4} = 0$ và $[Comable(F)]_{2,4} = 1$ nên S_8 khác rỗng. Do đó có thể chọn $h = 1$. Đảo bit $(2, 4)$ của F_1 để có F_1'.

+ Đối với F_2 : $SUM((F_2 \oplus K) \otimes W)(\text{mod}16) = 55(\text{mod}16) = 7$

Vì chuỗi 3 bit tiếp theo cần giấu là 111, nên ta có: $d = (1110) - 7 = 14 - 7 = 7$

Vì vậy, sẽ phải thay đổi các bit của F_2 để tăng trọng số lên 7.

Vì $[W]_{1,4} = 7$ và $[F_2 \oplus K]_{1,4} = 0$ và $[Comable(F)]_{1,8} = 1$ (điểm $(1, 4)$ trong F_2 tương ứng

với điểm $(1, 8)$ trong $Comable(F)$) nên S_7 khác rỗng. Do đó có thể chọn $h = 1$.

Đảo bit (1,4) của F_2 ta có F_2'.

+ Đối với F_3 : $SUM((F_3 \oplus K) \otimes W)(\text{mod}16) = 63(\text{mod}16) = 15$

Vì chuỗi 3 bit tiếp theo cần giấu là 011, nên ta có: $d = (0110) - 15 = 6 - 15 = -9$

Vì $S_{-9} = S_7(\text{mod}16)$ nên ta sẽ phải thay đổi các bit của F_3 để tăng trọng số lên 7.

Vì $[W]_{1,4} = 7$ và $[F_3 \oplus K]_{1,4} = 0$ và $[Comable(F)]_{5,4} = 1$ (điểm (1,4) trong F_3 tương ứng với điểm (5,4) trong $Comable(F)$) nên S_7 khác rỗng. Do đó có thể chọn $h = 1$.

Đảo bit (1,4) của F_3 ta có F_3'.

+ Đối với F_4 : $SUM((F_4 \oplus K) \otimes W)(\text{mod}16) = 53(\text{mod}16) = 5$

Vì chuỗi 3 bit tiếp theo cần giấu là 001, nên ta có: $d = (0010) - 5 = 2 - 5 = -3$

Vì $S_{-3} = S_{13}(\text{mod}16)$ nên ta sẽ phải thay đổi các bit của F_4 để tăng trọng số lên 13.

Vì $[W]_{3,3} = 13$ và $[F_4 \oplus K]_{3,3} = 0$ và $[Comable(F)]_{7,7} = 1$ (điểm (3,3) trong F_4 tương ứng với điểm (7,7) trong $Comable(F)$) nên S_{13} khác rỗng. Do đó có thể chọn $h = 1$.

Đảo bit (3,3) của F_4 ta có F_4'.

Ảnh F' được tạo thành từ việc ghép 4 khối điểm ảnh F_1', F_2', F_3', F_4' như sau.

Bước 5: Tính các tổng $SUM((F_i' \oplus K) \otimes W)$

$SUM((F_1' \oplus K) \otimes W) = 60$

$SUM((F_2' \oplus K) \otimes W) = 62$

$SUM((F_3' \oplus K) \otimes W) = 70$

$SUM((F_4' \oplus K) \otimes W) = 66$

Vì tất cả các F_i' đều không hoàn toàn đen và không hoàn toàn trắng và các tổng trên đều chẵn nên ta thu được các bit đã giấu như sau:

F'

0	1	0	1	0	1	1	1
1	0	0	1	1	1	1	1
1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0
1	1	1	1	1	0	0	0
0	0	1	1	1	0	1	0
1	1	0	1	0	1	0	1
1	0	1	1	0	1	1	1

$SUM((F_1' \oplus K) \otimes W)(\text{mod}16) = 60(\text{mod}16) = 12$, suy ra các bit đã giấu là $12/2 = 6 = 110$.

$SUM((F_2' \oplus K) \otimes W)(\text{mod}16) = 62(\text{mod}16) = 14$, suy ra các bit đã giấu là $14/2 = 7 = 111$.

$SUM((F_3' \oplus K) \otimes W)(\text{mod}16) = 70(\text{mod}16) = 6$, suy ra các bit đã giấu là $6/2 = 3 = 011$.

$SUM((F_4' \oplus K) \otimes W)(\text{mod}16) = 66(\text{mod}16) = 2$, suy ra các bit đã giấu là $2/2 = 1 = 001$.

Gộp lại ta có 110 111 011 001. Đây chính là chuỗi bit B ban đầu.

3.4. Kết quả thực nghiệm

Chúng tôi đã cài đặt lược đồ Teng-Pan và lược đồ cải tiến và thực hành một số thí nghiệm. So sánh các kết quả nhận được, chúng tôi thấy về chất lượng của các ảnh chủ sau khi giấu và khả năng giấu (độ dài tối đa của chuỗi bit cần giấu) của hai lược đồ là như nhau. Khi thử nghiệm với các ảnh chủ lớn, chúng tôi nhận thấy tốc độ thực hiện của lược đồ cải tiến nhanh hơn so với lược đồ nguyên bản. Điều này có được là nhờ việc thay thế ma trận khoảng cách bằng ma trân có thể đảo trong lược đồ cải tiến.

4. KẾT LUẬN

Trong bài báo này, chúng tôi đã trình bày một lược đồ giấu tin trong các ảnh hai màu được cải tiến từ lược đồ của Tseng-Pan trong [1]. Lược đồ này có thể giấu nhiều nhất là $\lfloor \log_2(mn+1) \rfloor$ - 1 bit trong mỗi khối ảnh chủ $m \times n$ bit mà chỉ cần đảo không quá hai bit của khối ảnh này. Các bit của khối ảnh chủ *có thể đảo được* phải có ít nhất một bit lân cận có giá trị khác với giá trị của chúng. Tính chất này được xét thông qua một ma trận được gọi là ma trận bit có thể đảo. Vì vậy, việc giấu dữ liệu vào các khối ảnh sẽ rất khó phát hiện được. Tuy nhiên để đạt được điều đó, chúng ta đã phải bỏ qua các khối mà trong đó không tìm được các bit có thể đảo và các khối toàn đen hoặc toàn trắng. Bạn đọc có thể dễ dàng nắm bắt được các bước thực hiện của lược đồ thông qua một ví dụ minh hoạ. Chúng tôi đã cài đặt lược đồ Tseng-Pan và lược đồ cải tiến. Kết quả thử nghiệm đối với một số ảnh chủ và các cỡ khối khác nhau cho thấy lược đồ này rất hữu hiệu khi ảnh chủ không có các mảng toàn đen hoặc toàn trắng và cỡ của khối là 12×12 hoặc lớn hơn. Lược đồ cải tiến chạy nhanh hơn so với lược đồ nguyên bản khi thử nghiệm với các ảnh chủ cỡ lớn.

Lược đồ Tseng-Pan và lược đồ cải tiến đương nhiên có thể áp dụng cho ảnh màu và ảnh đa cấp xám bằng cách chọn ra các bit ít quan trọng nhất của mỗi điểm ảnh để tạo thành một ảnh đen trắng và giấu dữ liệu vào ảnh này.

Nếu áp dụng tốt thuật toán này cho ảnh màu thì có thể nói thuật toán đã đạt được yêu cầu cơ bản của một ứng dụng giấu tin mật đó là đảm bảo tính ẩn của thông tin giấu nhờ không làm ảnh hưởng nhiều tới chất lượng ảnh chủ, số lượng thông tin giấu cao và độ an toàn của thuật toán.

TÀI LIỆU THAM KHẢO

[1] Yu Chee Tseng, Hsiang Kuang Pan, Secure and Invisible Data Hiding in 2- Color Images, *INFORCOM* 2001 , 887–896.

[2] Yu Yuan Chen, Hsiang Kuang Pan, and Yu Chee Tseng , A secure data hiding scheme for two-color images, *Fifth IEEE Symp. on Computer and Communication* (2000) 750–755.

[3] Báo cáo tổng thuật về các kỹ thuật giấu tin, Phòng Cơ sở dữ liệu và lập trình, Viện CNTT, Báo cáo của Đề tài cấp Viện KH&CN VN "Giấu tin trong ảnh", 2004.

[4] Min Wu, Edward Tang, Bede Liu, *Data Hidung in Digital Binary Image*, Princeton Summer Institute, 2000.

[5] W. Bender, D. W. Bender, D. Gruhl, N. Morimoto, and A. Lu, Techniques for data hiding, *IBM System Journal* **35** (NOS 384) (1996).

Nhận bài ngày 22 - 12 - 2003
Nhận lại sau sửa ngày 31 - 10 -2005