# A Text Steganography Method Based on Markov Chains

A. N. Shniperov, K.A. Nikitina

Siberian Federal University (SibFU), Krasnoyarsk, 660041Russia

E-mail: ashniperov@sfu-kras.ru

**Abstract**. A new method of text steganography based on Markov chains of different orders that allows the introduction of hidden information in texts is presented together with test results of a software solution which generate texts with a good approximation to the natural language model.
*Keywords*: text steganography, Markov process, automatic text generation, text naturalness.

## INTRODUCTION

Steganography is a science of a hidden data transmission by the concealment of the fact of data transfer. Currently, the problem of steganographic data protection from unauthorized access is extremely urgent [1]. However, most studies in this area are focused either on a concept of hidden information embedded in multimedia containers (images, audio and video files) of different formats, or aimed at the use of telecommunication networks (network steganography). At the same time, the development of linguistic steganography, which uses text information as a container, has received too little attention. This is explained by the fact that the steganographic methods based on the embedding of hidden information in media files, as well as in telecommunication networks, in fact, are not suitable when using natural language texts as a steganographic container.

Nevertheless, methods of text steganography can be widely used, since, according to statistics, it is textual information that has the highest transmission intensity [2]. The advantage of text containers over other media containers lies in the fact that methods of analysis of text files for the presence of hidden information have not been fully implemented at present [3]. At the same time, there are many algorithms designed exclusively for text containers, a description of which can be found, for example, in [4, 5].

Currently, there are a lot of methods of text steganography based on the use of Markov models. For example, in [2, 6] input data is used for text generation using Markov chains. However, the proposed models are greatly simplified in order to facilitate calculation, since it is assumed that all probabilities of transition from a given state to any other are equal.

The submitted paper is based on the studies of [7], in which the probabilities of transition of one word to another are maintained with sufficiently fair accuracy. The novelty of the method presented in this paper is that the proposed steganographic method is based on higher-order Markov processes (second and third) and also allows for working with Russian texts.

## 1. DESCRIPTION OF THE STEGANOGRAPHIC METHOD

The basic idea of the developed method is text generation (a steganographic container) on the basis of an available Markov chain and a concealed message. A Markov chain is constructed in advance using a text pattern, which has both the sender and the recipient. The text pattern is a text composed in a natural language. A generated text (a full steganographic container) may reflect a common meaning, at the same time each of its sentences will quite reliably repeat syntactically and grammatically some blocks of the text pattern.

The main stages of the proposed steganographic method are the following:

1. On the basis of some text pattern, hereinafter referred to as an empty container *C*, a transition probability matrix P for a Markov process is constructed, that corresponds the text *C*. A transition matrix element $P[i][j] = p_{s_i s_j} = \mathbb{P}\left(s_i \mid s_j\right)$ is the probability of transition from a state $s_i$ to the state $s_j$.

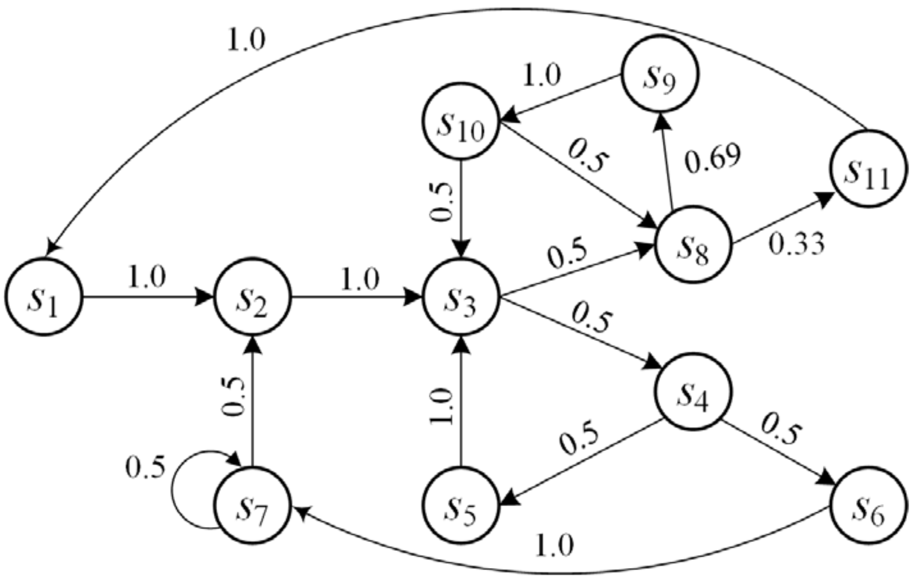

Fig. 1. A direct weighted graph G, corresponding to the transitional matrix in Table 1

Note, that the transition matrix P reflects the transition probability from each possible state $s_i$ to any other state. In general it can be represented as follows:

$$P = \begin{pmatrix} p_{s_1 s_1} & p_{s_1 s_j} & \cdots & p_{s_1 s_n} \\ p_{s_i s_1} & p_{s_i s_i} & \cdots & p_{s_2 s_n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{s_n s_1} & p_{s_n s_j} & \cdots & p_{s_n s_n} \end{pmatrix}$$

Under the state $s_i$ in terms of this work, we understand individual words (any sequence of characters between separators -- spaces and line breaks) or phrases (two words placed one after another through the separator).

The probability of the transition from a state $s_i$ to the state $s_j$ is a ratio of the number of times that the state $s_j$ immediately follows the state $s_i$ in the empty container $C$, to the number of separate iterations of the state $s_i$, i.e.,

$$p_{s_i, s_j} = \frac{n_{s_i s_j}}{n_{s_i}}, \qquad (1)$$

where $n_{s_i}$ is the number of iterations of the word in the text, and $n_{s_i s_j}$ is the number of iterations of the phrase $s_i s_j$.

As an example, without loss of generality, we can consider a transition matrix for a Markov process of a text *C*.

| | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ | $s_8$ | $s_9$ | $s_{10}$ | $s_{11}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $s_1$ | 0 | 1.0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $s_2$ | 0 | 0 | 1.0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $s_3$ | 0 | 0 | 0 | 0.5 | 0 | 0 | 0 | 0.5 | 0 | 0 | 0 |
| $s_4$ | 0 | 0 | 0 | 0 | 0.5 | 0.5 | 0 | 0 | 0 | 0 | 0 |
| $s_5$ | 0 | 0 | 1.0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $s_6$ | 0 | 0 | 0 | 0 | 0 | 0 | 1.0 | 0 | 0 | 0 | 0 |
| $s_7$ | 0 | 0.5 | 0 | 0 | 0 | 0 | 0.5 | 0 | 0 | 0 | 0 |
| $s_8$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.67 | 0 | 0.33 |
| $s_9$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1.0 | 0 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **$s_{10}$** | 0 | 0 | 0.5 | 0 | 0 | 0 | 0 | 0.5 | 0 | 0 | 0 |
| **$s_{11}$** | 1.0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Table 1. An example og the transition matrix P for a Markov process

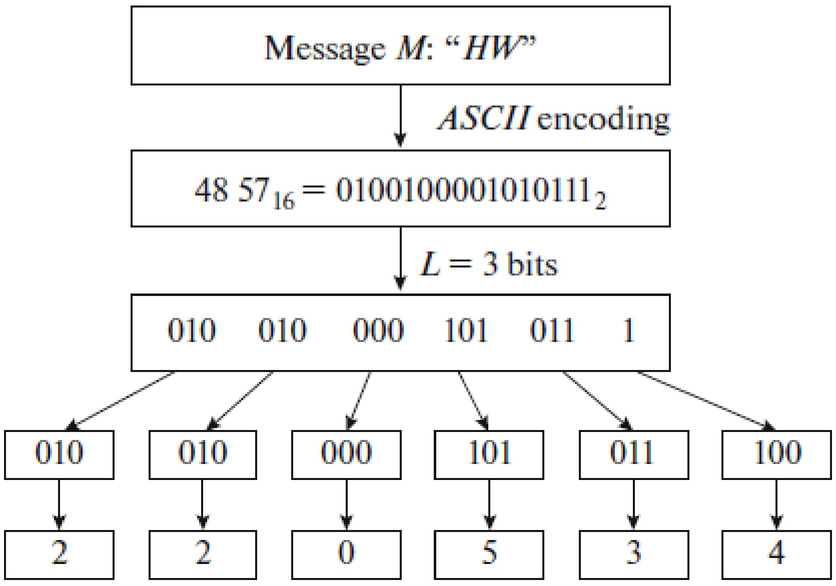

Fig. 2. An example of text encoding $M \to \tilde{M}$

This Markov process can be also represented as a weighted directed graph (see Fig. 1).

$$G = (V, E) \qquad (2)$$

when the nodes of the graph $V = \{s_1, s_2, \ldots, s_N\}$ are the states, the edges $E = \{(s_i, s_j)\}$ are possible transitions from one state to another. The weight of the edge is the probability of corresponding transition.
Note, that to eliminate "dead-end" states (those, for which all transition probabilities are equal to zero, and hence, the generation process stops), it is assumed that immediately after the end of the text *C* a Markov process begins again, with the latest states transferring further in the initial ones. Thus, in the above example the transition from the latest state of the text $s_{11}$ to the initial state $s_1$ with the probability $p_{s_{11}s_1} = 1.0$ was artificially added (initially all the transition probabilities from the state $s_{11}$ were equal to zero, since this is the latest state of the text).

2. Of all possible nodes of the graph *G* an initial node $s_k \in V$ is selected, which will be a key K to the steganographic system. The initial node (the initial state) is selected in such a way that its final punctuation mark turns out to be the last character, followed by the beginning of a new sentence.
Note, that the sender and the receiver have exchanged in advance the empty container *C* with the steganographic system key *K*.

3. An embedded hidden message M is pre-encoded in a decimal notation $M \to \tilde{M}$. Firstly, for this purpose, using its ASCII symbols, the message is converted into a binary notation and then divided into bit blocks of a fixed length L. Further, each of them is converted into a decimal notation in a range of $\left[0, 2^L - 1\right]$. Figure 2 represents an example of the line $M = \{m_1, m_2\} = \{H, W\}$ decimal encoding (for L = 3 bits) $\tilde{M} = \{\tilde{m}_1, \tilde{m}_2, \ldots, \tilde{m}_6\} = \{2,2,0,5,3,4\}$.

4. For a first decimal number $\tilde{m}_1 \in \tilde{M}$, a sequence of states $S_1 \subset V$ is generated on the basis of the transition probability matrix P by traversing of the graph nodes $G = (V, E)$ starting from the selected initial node $K = s_k$. The generation is performed as follows.
Let's write an interval $[a, b]$, which spans the embedded number $\tilde{m}_1$ initially $a = 0$ and $b = 2^L - 1$. In this case the interval length will be $len_{ab} = b - a + 1$. In the process of embedding the interval $[a, b]$ is broken down into subintervals proportionally to all the non-zero probabilities of transition from the current state $s_k$:

$[a,b]=[a,a_1]\cup[a_1,a_2]\cup\ldots\cup[a_t,b]$: where $len_{a_ia_j}=p_{s_ks_j}\cdot len_{ab}$ and the subinterval $[a_i,a_j]$ is selected, containing the embedded number $\tilde{m}_1\in[a_i,a_j]$.
At this point a condition $s_j$, which corresponds to the transition of the selected subinterval, becomes the current state and recorded in the full container and the selected subinterval $[a_i,a_j]$ becomes the current interval. Further, the process is iteratively repeated, i.e., again there is a division of the current interval (proportional to the nonzero probability of transition from the current state) and the selection of the subinterval containing $\tilde{m}_1$. The iterations are repeated until the current intervaling becomes impossible, i.e., $a=b=\tilde{m}_1$.

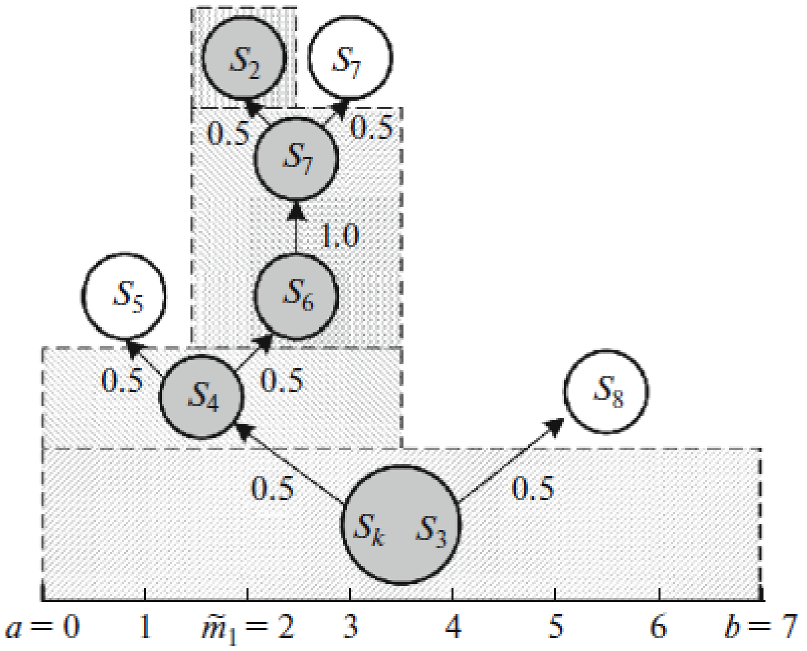

**Fig. 3.** Illustration of a sequence of states generation algorithm for the first decimal $\tilde{m}_1=2$ in the message

As an example, let us take for the initial node of the graph *G* the initial state $K=s_k=s_3$ (see Fig. 1) and the embedded number $\tilde{m}_1=2$. For the selected bit $L=3$, the initial interval is [0, 7]. A generation algorithm is shown in Fig. 3. The generation result will be a sequence of states $S_1=\{s_4,s_6,s_7,s_2\}$.

5. Similarly, for each of the remaining decimals $\tilde{m}_i\in\tilde{M}$ the respective sequence of state $S_i$ is generated, at the same time the latest state of the previous generation $S_{i-1}$ is selected as the initial state. In the case under study $s_2\in S_1$.

Thus, the resulting sequence of states $S=S_1\cup S_2\cup\ldots\cup S_h$, where *h* is a number of decimals of the encoded message $\tilde{M}$ forms a full steganographic container, which in fact will be a text, which also repeats some statistical features of the text of the empty container *C*.
A message *M* extraction algorithm from the full container *S* is similar to the embedding algorithm. As noted earlier, it is assumed that the empty container C is known to the receiver, which means that he can build for it the transition probability matrix *P* and the graph *G*. In addition, the receiver should also be aware of the initial state $K=s_k$, which means that to extract the characters $\tilde{m}_i\in\tilde{M}$ it is necessary to invert the embedding algorithm (see item 4) and use the sequence of states to restore $\tilde{m}_i$ a character of the encoded message. This occurs by similarly dividing the initial interval $[a,b]$ into parts in accordance with the probability of a state transition until it is reduced to $a=b=\tilde{m}_i$. Its inverse decoding is performed after the restoration of the entire $\tilde{M}\rightarrow M$ (see item 3).

## 2. THE USE OF HIGHER ORDER MARKOV PROCESSES

Second- and third-order Markov processes take into account two or three previous states rather than one. In this case, the transition Markov behavior matrix is not square, but vertical: it contains recorded transition probabilities of all possible pairs (or, respectively, triples) of states to all possible states (set states).

The dimension of the transition matrix is expressed by the formula $N^d \times N$, where $N$ is a number of unique states in the text, $d$ is a Markov process order, in this paper it is $d \in \{1,2,3\}$. The probability of the transition from the pair of states $s_i s_j$ to the state $s_k$ can be calculated by a formula similar to the formula (1):

$$p_{s_i,s_j,s_k} = \frac{n_{s_i s_j s_k}}{n_{s_i s_j}} \qquad (3)$$

where $n_{s_i s_j s_k}$ is a number of times a combination of states $s_i s_j s_k$ is met in the text.
Let us consider differences in the process of embedding of a hidden message for higher order Markov processes by an example of the second order. A key in this case will be a set of two states $K = s_k s_l$. An interval $[a,b]$ is divided into subintervals proportional to all possible nonzero probabilities of transition from the current set of states $s_k s_l$ to other (set) states. After the subinterval, which contains a decimal number of the embedded message block, is selected, the corresponding state $s_t$ is added to the full container *S*, and the current state becomes the set of states $s_l s_t$, composed of the second state of the previous current set and a new state.
When extracting, similarly, the current set always consists of a number of states equal to the Markov process order, each following current set is composed by dropping a first state of the set and adding a new state to the end. The process of subintervaling is not changed.

## 3. STEGANOGRAPHIC METHOD EXPERIMENTAL RESULTS AND ANALYSIS

The designed steganographic method was algorithmically implemented as a software solution for a Windows OS, written in C ++. This software solution was used for various experiments and the adjustment of conversion modes. During the experiments we used many various texts, empty containers *C* of the various kinds of content, as well as many messages *M* of different length, randomly generated from the alphabet characters, numbers, and punctuation symbols. The experiments were carried out with respect to the embedding methods based on the $d \in \{1,2,3\}$ order Markov processes.
Examples of the generation of a full container *S*, containing a hidden four-byte message *M* with arbitrary random characters are shown in Table 2. As an empty container *C,* we chose a book of Augustus Brown 'Why Pandas Do Handstands and Other Curious Truths about Animals' containing $V_C = 34668$ words.
The software we developed makes allowance for configuring the following parameters of the steganographic method:

- A Markov process order $d \in \{1,2,3\}$;
- A number of words in the state *s*: 1 or 2 (only for *d=1* order processes);
- A key *K* (a natural number on which basis the initial state is selected);
- A block *L* size, for encoding of $M \to \tilde{M}$.

| Program operation mode | Capacity |
|---|---|
| A first-order Markov process, a one-word state | 0.5–9% |
| A second-order Markov process, a one-word state | 0.007–2% |
| A third-order Markov process, a one-word state | 0.003–0.1% |
| A first-order Markov process, a two-word state | 0.003–0.5% |
| A first-order Markov process, a two-word state (for messages of recommended length) | 0.2–0.3% |

**Table 3.** Capacity of full containers *S* for different modes of the program operation

According to the test results of the embedding message method (with different parameters) the following parameters of empty and full containers were assessed:

- A volume V, the number of words in the text (for the empty containers the volume $V_C$ ranged from 414 to 178 214 words);

- Diversity D, the number of unique words in the text,

- Repeatability $R = \frac{V}{D}$ (for empty containers *C* from 1.45 to 6.48);
- Duplication, the number of duplicate keywords in the text.
- Vapidity, the percentage of insignificant words to the the total number of words in the text.

In the experiments, the following features of the system were identified:

(1) When the length a concealed message $len_M$ is significantly greater than the block *L* length, the volume of the full container $V_S$ does not depend on the selected length of the block *L*.

(2) The volume of a full container $V_S$ decreases linearly as the volume of the empty container $V_C$ increases. The repeatability $R_C$ also decreases as the volume $V_C$ increases, at the same time the repeatability $R_C$ and $R_S$ levels off when the volume is less than $V_C$ (in case of the first-order Markov process, the repeatability levels off when $V_S \approx \frac{1}{32} V_C$, in the case of the second-order process, when $V_S \approx \frac{1}{8} V_C$, and in the case of the third-order process, when $V_S \approx \frac{1}{2} V_C$).

(3) When the length of a concealed message $len_M$ is not less than the length of the block *L*, the volume of the full container $V_S$ is linearly dependent on the length of the message $len_M$.

(4) A number of unique words in the output container $D_S$ also increases linearly as $len_M$ increases to $D_C$ of the empty container.

(5) To determine a recommended length of a message $len'_M$ for a particular volume of the empty container $V_C$, when $V_S = q \cdot V_C$, where *q* is a certain part of the empty container, a special software module was developed. After the experiments carried out on various empty containers and their parts, when $q \in \{80\%, 90\%, 100\%\}$, it was found that the dependence of the recommended message length $len'_M$ on the volume of the empty container $V_C$ is almost linear.

(6) After a large number of experiments, it became possible to calculate an average capacity of the hidden message *M* within the proposed method, i.e., the relationship between the concealed message *M* volume and the volume of the full container $V_S$. The values are presented in Table 3. It may be concluded that the use of the third-order Markov processes significantly reduces the method's capacity (less than 0.1%), which makes its application difficult.

(7) The duplication and vapidity of the container do not depend (within statistical error) on the embedding parameters being equal to the corresponding parameters of the empty container. This can be explained by the fact that both parameters are determined by counting the usage frequency of certain words, and the text generation algorithm, based on Markov chains, saves with some accuracy the usage frequency of all words.

## CONCLUSIONS

The paper presents a new method of text steganography based on Markov chains of different orders, which differs from the existing ones in the more accurate consideration of transition probabilities from one word to another, the ability to use different languages, as well as the use of higher-order Markov chains.

The statistical estimation of the main full container characteristics is carried out. In view of this research, the steganographic method criteria are formalized.

On the basis of the analysis, it was concluded that it is reasonable to use the proposed method in the following mode: concealment of relatively short messages (up to 500 characters) on the basis of a second-order Markov process for a one-word state, or on the basis of a first-order Markov process for a two-word state in sufficiently long texts of empty containers (of 10,000 words). In this case, a full container is similar to the empty one in volume and repeatability. Its capacity is quite large and the consistency of the texts with the expert analysis is satisfactory.

It is important to note that at the moment a universal method of text file computer-aided analysis for the presence of hidden information has not been found, as well as the problem of checking texts for natural origin (whether it is a man-made text or not) has not been solved. Statistical metrics for the evaluation of text naturalness used in many cases do not always provide identification of hidden channels of data transfer using text steganography. In this connection, the effectiveness of the proposed method, given the selection of optimal parameters, is quite high, especially in intensive workflow conditions.

REFERENCES

1. Konahovich, G.F. and Puzyrenko, A.Y., *Komp'yuternaya steganografiya. Teoriya i praktika* (Computer Steganography. Theory and Practice), Moscow: MK-Press, 2006.
2. Dai, W., Yu, Y., and Deng, B., BinText steganography based on Markov state transferring probability, *2nd International Conference on Interaction Sciences: Information Technology, Culture and Human,* 2009, pp. 1306–1311.
3. Meng, P., Huang, L., Chen, Z, Yang, W., and Li, D., Linguistic steganography detection based on perplexity, in *MultiMedia and Information Technology,* 2008, pp. 217–220.
4. Nechta, I.V., An effective method of steganalysis based on data compression, *Vestn. SibGUTI,* 2010, no. 1, pp. 50–55.
5. Nechta, I. and Fionov, A., Applying statistical methods to text steganography, *Applied Methods of Statistical Analysis. Simulations and Statistical Inference,* Novosibirsk, 2011, pp. 278–284.