A Framework for Designing Provably Secure Steganography

Guorui Liao, Jinshuai Yang, Weizhi Shao, and Yongfeng Huang

Tsinghua University

Abstract

Steganography is a technique to transmit secret messages over a public channel so that the very existence of these secret messages can not be detected. In this field, provably secure steganography based on shared white-box samplers is a major focus due to its capability to construct secure and efficient steganographic systems on various practical channels. However, designing a novel provably secure steganography scheme remains challenging, since the scheme must maintain a nearly identical sampling distribution to any given discrete distribution while embedding secret information. Currently, there are only a few provably secure steganography schemes available, which significantly limits both practical application and theoretical research. In this paper, we propose a framework for designing provably secure steganography, with the universal security proof for schemes derived from this framework. This framework decomposes the overall complex design into three sub-processes that can be relatively easily achieved, namely Probability Recombination Module, Bin Sampling and Uniform Steganography Module. With this framework, we present several new provably secure steganography schemes and demonstrate that the recent work, Discop(base), is also encompassed by this framework. Additionally, guided by this framework, we have identified several schemes that are theoretically optimal or very effective under specified metrics and validated their effectiveness through experimental verification.

1 Introduction

Privacy security, particularly the right to unmonitored communication in cyberspace, suffers severely from the proliferation of surveillance and censorship [16, 17, 25, 33]. In response to these rising cyberspace threats, encryption-based techniques have been widely used in cyberspace communication systems. They aim to protect the privacy of transmitted content by thwarting content censorship through end-to-end encrypted communication [6, 11], or to anonymize identities to circumvent surveillance, as seen with technologies like Tor [13, 29].

However, the protection offered by these technologies has its limits. Traffic from Tor is easily identifiable [23, 24, 31], which can lead authorities to easily detect and block such communications outright. Additionally, in settings where strict control is imposed, there may be a requirement for only plaintext transmissions, disallowing any encrypted communication if the content cannot be verified. This issue arises because these technologies do not conceal the fact that a communication is taking place, exposing it to potential disruptions. Furthermore, they often imply the existence of sensitive information, potentially leading to enhanced monitoring.

Thus, to counteract stringent content censorship, steganography offers a solution: hiding the very existence of secret messages during their transmission so that a third party (especially the monitor) cannot detect them [18]. One method achieves this by disguising secret messages as seemingly normal data, termed *stegotext*, within their respective communication channels, such as texts [36, 40] in natural language channels or images [8] in visual channels. The actually normal data, such as appropriate text in a talking scene or legitimate network traffic, is termed *covertext*. Informally, this process can be described as generating/sampling *stegotext* to act as the carrier for secret messages, thus called generative steganography [1]. For a sufficiently normal-looking stegotext that cannot be distinguished from covertext, censorship continues as usual, yet without any detection of abnormalities. Additionally, there is no effective way to only terminate the steganographic communication without also terminating all normal communication in the channel, thus circumventing the problems mentioned earlier.

A representative application example of generative steganography is its use within natural language channels [36, 40, 42, 43], driven by the desire to securely and efficiently embed secret information within the most common human communication. In practice, security often solely involves avoiding unusual linguistic expressions or vocabulary. However, relying solely on the empirical security measures can hardly be convincing and is easily detected, whether for text or other carriers [20, 37, 41, 44].

In order to implement secure steganography, researchers have proposed a variety of steganography models under different settings to describe steganography and its security [7, 18, 19]. Researchers consistently describe the process of generating covertext (or stegotext) as being accomplished through a sampler (or steganographic sampler) within the channel, making the nature of the sampler a crucial setting. In modern days, with the development of AI-generated content (AIGC), white-box samplers with explicit discrete distributions have emerged (e.g., large language models for natural language channels [3, 14, 21, 34], simulations for network traffic channels [26]) and are considered capable of effectively sampling covertext within the channel. Therefore, considering practical applications, researchers are currently focusing on a provably secure symmetric steganography model under the white-box sampler setting [22]. This steganography model is often referred to simply as **Provably Secure Steganography** in some literature [12, 27], a term we will adopt in this paper.

However, constructing schemes within this steganography model remains a difficult task. Each scheme must meet rigorous standards of *security* and *universality*. This means a provably secure steganography scheme must not only achieve computational indistinguishability against chosen hiddentext attacks by *ppt.* (probabilistic polynomial-time) adversaries but also maintain applicability across various discrete distributions determined by white-box samplers within the channel. This dual requirement makes designing such schemes one of the most challenging theoretical problems in the field of steganography. Notably, such schemes are applicable in any channel equipped with a white-box sampler, thereby representing some of the most important and universally relevant issues in steganography. Currently, only a few schemes like *Meteor* [22] and *Discop* [12] have managed to meet these criteria based on *secure PRG* (Pseudorandom Generators) and achieve practical implementation. Due to the numerous factors influencing steganography, these schemes do not perform optimally in all metrics and situations, such as capacity within uniform distributions. Moreover, the limited variety of available scheme options fails to meet the diverse demands of practical applications. This scarcity also hinders further exploration into the theory of such steganography model.

In this paper, we introduce a framework for constructing provably secure steganography schemes. This framework allows us to easily construct provably secure steganography schemes based on two simple modules. We recognize that this proposed framework does not encompass all provably secure steganography schemes, but this proposed framework still enables us to develop schemes that excel or achieve optimal in some specific metrics of common concerns.

The observation is that the difficulty in constructing a provably secure steganography scheme is partly due to the challenge of achieving validity across different distributions. If we can, by some means, transform the task of constructing such a scheme universally applicable to any distribution into constructing one applicable only to specific distributions, we can then focus our efforts on building steganography with security tailored for those specific distributions, which is easier. This is the function of the two main modules of our framework. The first module simplifies the task through probability recombination. In the second module, we only need to design a steganography scheme for the uniform distributions. These two modules are interconnected through a sampling operation. We provide a universal security proof for steganography schemes produced by this framework under the assumption of *secure PRG*. Additionally, we give some different module constructions that can be combined into various provably secure steganography schemes and demonstrate that the construction equivalent to the recent scheme Discop(base) can be derived within our framework. Finally, under the guidance of our framework, we construct provably secure steganography schemes that excel in terms of capacity, stability, and time complexity.

Contributions The contributions of this work can be summarized as follows:

- We propose a novel framework for designing new provably secure steganography schemes which makes the design process easier and we provide a security proof, demonstrating that all steganography derived from our framework are provably secure steganography schemes.
- We introduce many novel provably secure steganography schemes with this proposed framework. Additionally, we also clarify that the recently proposed scheme Discop(base) can be derived from our framework.
- Under the guidance of our framework, we have developed provably secure steganography schemes optimized for key metrics such as capacity, stability, and time complexity. We have both theoretically derived and experimentally validated the performance of these schemes, demonstrating that they achieve or closely approach optimal performance in certain scenarios.

Organization Section 2 details the background of provably secure steganography, including its history, definition, and related works in practice. Section 3 describes the proposed framework and the accompanying security proof. Section 4 introduces typical derivative constructions based on our framework and explains why Discop(base) can be considered a construction derived from our framework. Section 5 discusses the design of schemes tailored to key performance metrics, such as capacity, stability, and time complexity. This section also includes comparisons with existing schemes across these metrics. Finally, Section 6 presents our conclusions.

2 Background and Related Work

2.1 Classic Steganography Model

Information-Theoretic Model The earliest formal definition of a steganography model comes from Cachin [7], who defined "perfectly secure steganography" and "ε-secure steganography" based on information-theoretic security. In this model, both covertext P_C and stegotext P_S are treated as random variables, enabling the problem of distinguishing between them to be analyzed using hypothesis testing theory.

In this model, a scheme satisfying

$$KL(P_C||P_S) = \sum_{v \in V} P_C(v) \log \frac{P_C(v)}{P_S(v)} \leq \varepsilon$$

is termed "ε-secure steganography". Here, $KL(P_C||P_S)$ denotes the Kullback-Leibler divergence (KLD). When $\varepsilon = 0$, the scheme is termed "perfectly secure steganography", indicating that the distributions of the two random variables are identical, and thus indistinguishable to third parties.

Universal Steganography Models Like most cryptosystems, achieving a perfectly secure scheme in steganography is practically unrealistic; hence, steganography shifted towards definitions based on computational security. The fundamental intuition of these security definitions is that a *ppt.* (probabilistic polynomial-time) adversary cannot distinguish stegotext from covertext. Unlike most cryptosystems, however, stegosystems involve the nature of the communication channel itself and the interaction between the communicating entities and the channel.

Hopper *et al.* [18, 19] were the first to formalize these settings in steganography, defining the communication channel as a series of random variables and describing how communicating parties could interact with the channel through an oracle. In Hopper's setup, the monitor has complete knowledge of the communication channel, the information sender can only sample from the channel through an oracle, and the receiver has no prior knowledge of the channel. This is known as the universal steganography model.

Subsequent work has proposed many schemes building on this model [2, 4, 5, 35]. However, there are some significant limitations. These schemes, built on black-box samplers (oracle) and relying on the rejection sampling, often depend on strong assumptions: 1. Informative channel; 2. Unbiased hash function. In practice, it has been shown that such assumptions are not reasonable in some channels [22], and these schemes also have extremely low capacity. Additionally, obtaining an oracle is still a problem at that time, hence often impractical to apply.

2.2 Provably Secure Steganography

The first practical application of generative steganography came from the natural language processing (NLP) community [9, 15, 32, 40, 42, 43], where researchers sought to convey secret messages through natural language while ensuring normal linguistic expression. They used highly successful generative language models [3, 21, 28], which can predict the distribution of the next token (that is, punctuation mark, emoji, part of word, word and so on) based on a given sequence of tokens. Coupled with suitable mapping algorithms (where part of the tokens represent secret information '1', and others '0'), which serve as an alternative to random sampling, this method enables the transmission of secret messages in textual form.

However, these efforts primarily focused on fluid and reasonable linguistic expression without considering security aspects. In fact, most of these algorithms significantly alter the channel distribution, making them easily distinguishable by statistical and machine learning methods [37–39, 41, 44]. Therefore, this period's work can hardly be considered secure stegosystems.

Kaptchuk *et al.* [22] extended the ideas of the universal steganography model and re-formalized the intuitive concepts derived from the works of NLP researchers, which include the capability to acquire explicit distributions during the sampling process (white-box sampler), contrasting with the reliance on a black-box oracle that only allows sampling. Both the sender and receiver have access to these distributions. This steganography model, which leverages the symmetric access to channel distributions, is referred to as the provably secure symmetric steganography model, or simply **Provably Secure Steganography** [12]. And for completeness in our discussion, we include here the formal definitions provided by [22]:

2.2.1 Definition

A provably secure steganography scheme $\Sigma_{\mathcal{D}}$, under a covertext channel distribution $\mathcal{D}$, consists of a triple of possibly probabilistic algorithms. These are denoted as $\Sigma_{\mathcal{D}} = (\text{KeyGen}_{\mathcal{D}}, \text{Encode}_{\mathcal{D}}, \text{Decode}_{\mathcal{D}})$.

- **KeyGen**$_{\mathcal{D}}(1^{\lambda})$: An algorithm takes an input of arbitrary length λ and generates K, a key that will be used in the other two algorithms.
- **Encode**$_{\mathcal{D}}(K, M, \mathcal{H})$: An algorithm that accepts a key K, a secret message M, and the channel history $\mathcal{H}$. This algorithm returns a stegotext S which is a sequence of covertext symbols.
- **Decode**$_{\mathcal{D}}(K, S, \mathcal{H})$: An algorithm that accepts a key K, a stegotext S, and the channel history $\mathcal{H}$. This algorithm returns the secret message M on success or the empty string ε on failure.

Correctness A scheme must ensure correctness, meaning that, with only a negligible probability of error, an encoded message should be recoverable by the decoding algorithm:

$$\Pr[\text{Decode}_{\mathcal{D}}(K, \text{Encode}_{\mathcal{D}}(K, M, \mathcal{H}), \mathcal{H}) = M] \geq 1 - \text{negl}(\lambda).$$

Security A scheme must ensure security against *chosen hiddentext attacks*, meaning that, for all *ppt.* adversaries $\mathcal{A}_{\mathcal{D}}$, the following inequality holds:

$$\left| \Pr\left[\mathcal{A}_{\mathcal{D}}^{\text{Encode}_{\mathcal{D}}(K,\cdot,\cdot)} = 1\right] - \Pr\left[\mathcal{A}_{\mathcal{D}}^{O_{\mathcal{D}}(\cdot,\cdot)} = 1\right] \right| < \text{negl}(\lambda),$$

where $O_{\mathcal{D}}(\cdot,\cdot)$ is an oracle capable of randomly sampling from the distribution $\mathcal{D}$.

Notably, channel distribution $\mathcal{D}$ is used to describe the overall channel (a series of random variables). At each time step, we denote the corresponding distribution by $D \leftarrow \mathcal{D}(\mathcal{H})$ to distinguish it. Additionally, we refer to the symbols of covertext as *tokens*. Therefore, each D is essentially a discrete probability distribution defined over the vocabulary of tokens. In practice, the **Encode**$_{\mathcal{D}}$/**Decode**$_{\mathcal{D}}$ consists of sequential time-step encode and decode operations. Each time-step operation takes the time-step distribution D as input and outputs either the selected token or the secret bits.

As in most previous works [12, 35, 42], we assume that the secret message is a sufficiently long encrypted bit stream. This implies that each bit in the message can be considered independent and random.

2.2.2 Practical Schemes

Meteor Kaptchuk *et al.* [22] developed the first steganographic scheme—Meteor—that satisfies this definition. The scheme utilizes a secure Pseudo-Random Generator (PRG) to share random bits at each time step of steganographic sampling, acting as a mask for the secret information. This masked secret information is then treated as a random number for sampling and is decoded deterministically using a Ranged Randomness Recoverable Sampling Scheme, akin to arithmetic coding. This approach offers excellent practical security with small additional time overhead. However, there is still room for improvement in capacity. An improvement based on reorder (Meteor-reorder) significantly enhances capacity, but it also substantially increases the time required.

Discop Ding *et al.* [12] introduced a new scheme Discop(base), where at each time step of sampling, the sender and receiver construct "distribution copies". The secret message is then used to determine which "distribution copies" to use, and a shared random number generated by a *secure PRG* is employed to select a token within the chosen "distribution copy". The authors also employed an iterative method based on the Huffman tree to further enhance the capacity, referred to as Discop(Huffman). Experimental results demonstrated an exceptionally high utilization rate of entropy.

iMEC Another related steganographic scheme is iMEC [10],which explores how much information the sender and receiver can deduce about a fixed-length secret message through a chosen token and aims to maximize and accumulate this information until the secret message is fully determined. However, this method does not fully satisfy the definition of provably secure steganography, as it involves an unavoidable error rate related to the channel distribution, which may be non-negligible, especially when the channel length is limited. Besides, the computational efficiency of iMEC is extremely low, making it difficult to be practically implemented in a large token vocabulary situation. Despite these limitations, this approach still offers significant insights for the development of provably secure steganography.

3 Our Framework and Its Security Proofs

As previously mentioned, the challenges involved in constructing provably secure steganography schemes have resulted in a limited number of existing solutions. In this section, we introduce a framework designed to simplify this process.

One of the challenges in designing provably secure steganography schemes arises from the need for universality—that is, the scheme must be valid across the entire covertext channel distribution $\mathcal{D}$, where each time-step distribution D can be any arbitrary discrete distribution. Thus, a natural aspiration is to "simplify" the requirement for any arbitrary discrete distribution through some efficient algorithm to only need to accommodate some specific distributions. Generally, only constructing a scheme valid for specific distributions is much simpler than designing a universal one.

Our framework describes this "simplified" methodology and demonstrates how to construct a provably secure steganography scheme from both a "simplified" scheme and a steganography scheme valid only for uniform distributions. In our framework, the "simplified" methodology involves **Probability Recombination Module** and **Bin Sampling**, while the provably secure steganography scheme for specific distributions is described in **Uniform Steganography Module**, see Figure 1.

The main idea of the Probability Recombination Module at each time step is to transform a complex discrete probability distribution D into a collection of uniform distributions. Bin Sampling then select one uniform distribution from this using random numbers shared by communication entities. Finally, the Uniform Steganography Module needs only to implement a provably secure steganography scheme valid for the selected uniform distribution.

3.1 Probability Recombination Module

This module describes a class of schemes that decompose and recombine the probability values of any arbitrary discrete distribution into a set of non-overlapping uniform distributions.

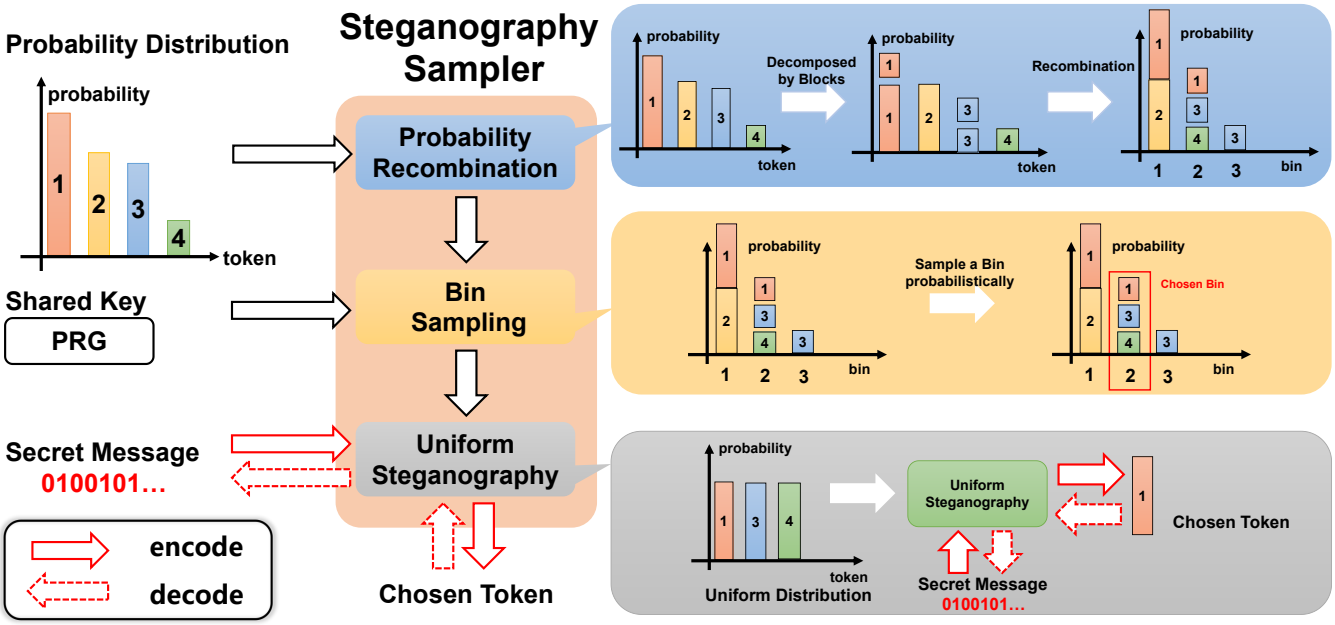

Figure 1: The illustration of the overall framework.

Specifically, it takes as input an arbitrary discrete distribution $D = [p_1, p_2, .., p_n]$ where $\sum_{j=1}^{n} p_j = 1$. In this module, the probability p_j of each token j is decomposed into several **non-negative** p_j^k, where $p_j = \sum_k p_j^k$ and each p_j^k is termed a **block** of token j. For example, if a token has a probability of 0.5, it could be decomposed into blocks 0.4 and 0.1. Note that, the decomposition can be performed arbitrarily. Subsequently, we combine blocks from different tokens into **bin**s according to the following principles:

1. Each block must be assigned to exactly one bin.
2. Blocks within a bin must have equal probability values.
3. No token can have more than one block in the same bin.

We denote the set of bins by $\mathcal{G} = \{G_i\}_{i \le N}$, where G_i is a certain bin. And if a block p_j^k of token j was assigned to the bin G_i, we express this relationship as $p_j^k \in G_i$. Thus the 3 principles above can be formulated as:

1. For any p_j^k there exists G_i such that $p_j^k \in G_i$.
2. For any i and $p_{j_1}^{k_1}, p_{j_2}^{k_2} \in G_i$, it holds that $p_{j_1}^{k_1} = p_{j_2}^{k_2}$.
3. For any i and $j, p_j^{k_1}, p_j^{k_2} \in G_i$ indicates $k_1 = k_2$.

Then, the output of the probability recombination module is the set of bins $\mathcal{G}$.

For ease of discussion, the probability of a bin $P[G_i]$ is defined as the sum of the probabilities of the blocks it contains, and we use $\mathcal{D}_\mathcal{G} = [P[G_1], P[G_2], ..., P[G_N]]$ to represent the distribution of these bins. This formulation is a valid distribution due to the first principle. Additionally, we denote T_i as the set of tokens from which a block is derived before assigned to G_i, and define $\mathcal{T}_\mathcal{G} = \{T_i\}_{i \le N}$.We can observe that the pair$(\mathcal{D}_\mathcal{G}, \mathcal{T}_\mathcal{G})$ also serves as an equivalent representation of $\mathcal{G}$.

For instance, consider a probability distribution with "A" at 0.5, "B" at 0.4, and "C" at 0.1. One feasible approach to recombination might involve splitting "A" into blocks of 0.4 and 0.1, while treating "B" and "C" as individual blocks. Consequently, the first bin could include the 0.4 block from "A" combined with the block from "B", and the second bin could consist of the 0.1 block from "A" combined with the block from "C". This configuration adheres to the three principles:

$$\textbf{Bin 1}: G_1 = [0.4_A, 0.4_B] \quad \textbf{Bin 2}: G_2 = [0.1_A, 0.1_C]$$

The resulting distributions for the bins and the corresponding token sets are

$$\mathcal{D}_\mathcal{G} = [0.8, 0.2], \mathcal{T}_\mathcal{G} = [\{A, B\}, \{A, C\}].$$

Through this module, we achieve a recombination of the original probability distribution D of tokens into a new probability distribution $\mathcal{D}_\mathcal{G}$ of bins. Moreover, by considering the blocks within each bin, we ensure a uniform distribution (due to the second principle).

We refer to a specific implementation that fulfills these requirements across all discrete distributions as a "Probability Recombination Scheme". Some implementations are introduced in Section 4.2.

3.2 Bin Sampling

The Bin Sampling takes as input the bins $\mathcal{G}$ from the Probability Recombination Module and samples a bin G_i as output, based on its distribution $\mathcal{D}_{\mathcal{G}}$. To facilitate this sampling, a *secure PRG* is employed, with the key shared between the sender and the receiver. This shared key ensures that at each time step, both parties can generate the same pseudo-random number in range $(0,1]$[1]. To determine which bin is sampled, the range $(0,1]$ is divided into intervals corresponding to the bins in $\mathcal{G}$ based on the probabilities in $\mathcal{D}_{\mathcal{G}}$. The pseudo-random number is then used to identify the bin by finding the interval it falls into. Consequently, both the sender and the receiver can determine which bin was sampled at each time step.

3.3 Uniform Steganography Module

We use the term "uniform steganography" to denote a weakened version of provably secure steganography. The only difference is that the conditions which previously had to be met for all discrete distributions now only need to be satisfied for uniform distributions. In other words, all the distributions D in the covertext channel distributions $\mathcal{D}$ are uniform distributions.

At each time step, the Uniform Steganography Module takes a bin G_i as input from the Bin Sampling. Given that the distribution of blocks in G_i can be considered as a uniform distribution, this module functions effectively under these conditions. Using the time-step encoding algorithm of a uniform steganography scheme, this module can take a secret message as input, steganographically sample a block p_j^k from G_i and then output the token j from which the block originated.

It also takes a token j as input, identifies the only block p_j^k that comes from j within the bin G_i (due to the third principle), and decodes the secret message using the time-step decoding algorithm of the same uniform steganography scheme.

For instance, consider the scenario where **Bin 1**: $G_1 = [0.4_A, 0.4_B]$ is sampled during Bin Sampling. This bin can then be viewed as a 2-uniform distribution [0.5,0.5] with the token set $\{A,B\}$. In this simplified uniform distribution, we could employ a straightforward steganography: choose token "A" when the secret bit is '0', and choose token "B" when the secret bit is '1', without distorting the original distribution.

3.4 Integration of Modules

At each time step, sequentially chaining a Probability Recombination scheme **PR**, Bin Sampling, and a Uniform Steganography scheme **US** (with the time-step encoding algorithm **USE** and decoding algorithm **USD**) yields the corresponding time-step encoding/decoding algorithm, which, when applied step-by-step across all time steps, forms the **Encode**$_{\mathcal{D}}$/**Decode**$_{\mathcal{D}}$ algorithm, shown as Algorithm 1 and Algorithm 2. Additionally, another intuitive diagram demonstrating this integration at each time step is provided in Figure 1.

[1]A finite-length bit stream can represent a fraction in binary.

Algorithm 1 Encode$_{\mathcal{D}}^{\mathbf{PR,US}}$

Input: Key K_{PRG} and K_{US}, Secret Message M, History $\mathcal{H}$
Output: Stegotext S

$S \leftarrow$ " "
PRG.set_key(K_{PRG})
while not the end of S **do**
  $D \leftarrow \mathcal{D}(\mathcal{H}||S)$ ▷ Time-Step Distribution
  $\mathcal{G} \leftarrow PR(D)$ ▷ Probability Recombination
  $G_i \leftarrow$ **Sample**$(\mathcal{G}, PRG)$ ▷ Bin Sampling
  $s,c \leftarrow$ **USE**(G_i, K_{US}, M) ▷ Uniform Steganography
  $M \leftarrow M[c:], S \leftarrow S||s$
end while
return S

Algorithm 2 Decode$_{\mathcal{D}}^{\mathbf{PR,US}}$

Input: Key K_{PRG} and K_{US}, Stegotext S, History $\mathcal{H}$
Output: Secret Message M

PRG.set_key(K_{PRG})
$n \leftarrow 0$
$M \leftarrow$ " "
while $n < |S|$ **do**
  $D \leftarrow \mathcal{D}(\mathcal{H}||S[:n])$ ▷ Time-Step Distribution
  $\mathcal{G} \leftarrow PR(D)$ ▷ Probability Recombination
  $G_i \leftarrow$ **Sample**$(\mathcal{G}, PRG)$ ▷ Bin Sampling
  $s \leftarrow S[n]$
  $M \leftarrow M||$**USD**(G_i, K_{US}, s) ▷ Uniform Steganography
  $n \leftarrow n+1$
end while
return M

Correctness Noting the symmetry, the chosen bin G_i at each time step is identical for both the encoding and decoding algorithms. Therefore, the overall correctness is guaranteed by the correctness of the uniform steganography scheme used.

3.5 Security Proofs

Before delving into formal proofs, let us gain an intuitive understanding of the security of this framework through an informal proof.

(Informal Proof) At each time step, the probability $P_{stego}[i]$ of token i being selected within our steganography framework can be calculated as:

$$
\begin{aligned}
P_{stego}[i] &= \sum_{G_j \in \mathcal{G}} P_{BS}[G_j] \cdot P_{US}[i \mid G_j] \\
&\approx \sum_{j:\, \exists k,\, p_i^k \in G_j} P[G_j] \cdot \frac{1}{|G_j|} = \sum_k p_i^k = p_i.
\end{aligned} \tag{1}
$$

Here, $P_{BS}[G_j]$ represents the probability of selecting bin G_j during the Bin Sampling, and $P_{US}[i \mid G_j]$ denotes the probability of selecting token i's block after G_j has been chosen during the Uniform Steganography Module. The approximation arises because Bin Sampling functions *similarly* to random sampling based on bin's probability, and Uniform Steganography Module *mimics* random sampling within a uniform distribution due to its security. The third equality follows from the definition of $P[G_j]$ given in Section 3.1.

Therefore, this steganographic sampler does not change the original distribution D at each time step. In addition, the parameters used for each time-step sampling are independent. This independence ensures that the overall channel distribution $\mathcal{D}$ remains unchanged, thus preventing an attacker from distinguishing between random sampling and steganographic sampling outcomes.

Formally, to demonstrate that the two distributions $\mathcal{D}$ and $\hat{\mathcal{D}}$, which are derived from $O_{\mathcal{D}}(\cdot,\cdot)$ and $\text{Encode}_{\mathcal{D}}(K,\cdot,\cdot)$ respectively, are computationally indistinguishable, we define a sequence of hybrid distributions as $\mathcal{D} := H_0, H_1, H_2, H_3 =: \hat{\mathcal{D}}$. The transformations between these hybrids are constructed as follows:

- H_0 is constructed by randomly sampling tokens from each time-step distribution D and concatenating them, which is exactly the sampling result of the channel distribution $\mathcal{D}$.
- H_1 differs from H_0 in the sampling method at each time step. Instead of randomly sampling a token from the time-step distribution D, H_1 is obtained by first employing the probability recombination module to generate $\mathcal{G}$, then sampling a bin according to its probability, and finally sampling a block within the bin based on its probability to determine the corresponding token.
- H_2 differs from H_1 in that the probabilistic random sampling of the bin is replaced with the Bin Sampling described in Section 3.2.
- H_3 is modified from H_2 by replacing the sampling of the block with Uniform Steganography Module, which corresponds to the $\text{Encode}_{\mathcal{D}}(K,\cdot,\cdot)$.

Note that $H_0 = H_1$ is directly proven by Equation 1 in the informal proof. If there exists an adversary $\mathcal{A}$ with a non-negligible advantage in distinguishing H_1 from H_2 , we can construct a polynomial-time adversary $\mathcal{B}$ distinguishing between the outputs of the oracle $O(\cdot)$, which produces true randomness, and the outputs of a *secure PRG*. Adversary $\mathcal{B}$ runs $\mathcal{A}$ as a subroutine and simulates responses for $\mathcal{A}$ following the behavior of H_0 or H_1, where the randomness for selecting the bin is either sourced from $O(\cdot)$ or the *secure PRG*. By making $\mathcal{B}$ output the same guess as $\mathcal{A}$, $\mathcal{B}$ has the same, non-negligible advantage. However, the assumption of a *secure PRG* ensures that no such adversary $\mathcal{B}$ exists, which implies that $\mathcal{A}$ also cannot exist. Similarly, if H_2 and H_3 were distinguishable, It would be trivial to construct an effective adversary with a non-negligible advantage in distinguishing the sampling results from a channel distribution $\mathcal{D}'$, where all time-step distributions are uniform, and the corresponding steganographic outputs. This would violate the security of the uniform steganography scheme used.

Thus, based on the assumption of *secure PRG* and the security of the uniform steganography scheme, we have:

$$
D := H_0 \overset{p}{=} H_1 \overset{c}{\approx} H_2 \overset{c}{\approx} H_3 =: \hat{D}.
$$

The security proof above confirms that any steganography scheme developed within this framework adheres to the corresponding security definition outlined in Section 2.2.1. Consequently, by employing any probability recombination scheme and uniform steganography within this framework, we can establish a provably secure steganography scheme.

4 Typical Implementations of the Framework

In this section, we present several implementations of probability recombination and uniform steganography schemes, which can be used to construct various provably secure steganography schemes within our framework. At the end of this section, we will also briefly discuss why recently proposed scheme Discop(base) [12] can be considered a specific case within our framework.

4.1 Notation

For clarity in the following discussion, we adopt the following notations: $U(n)$ denotes the uniform distribution with n tokens. $U(1,n)$ signifies the uniform distribution over the integers from 1 to n. The modulo operation is represented by " mod ", where "a mod b" is the remainder of a divided by b, with the result adjusted to range from 1 to b by default. The function floor(x) returns the largest integer less than or equal to x. The function int2bits(x, k) converts an integer x into its binary representation with k bits, for example int2bits(2, 2) = '10'. Conversely, bits2int(bits) converts a binary string bits back into an integer. The notation $[:n]$ refers to the first n items of a list. The operator $||$ represents the concatenation of strings. The symbol $|\cdot|$ represents the number of elements in a set.

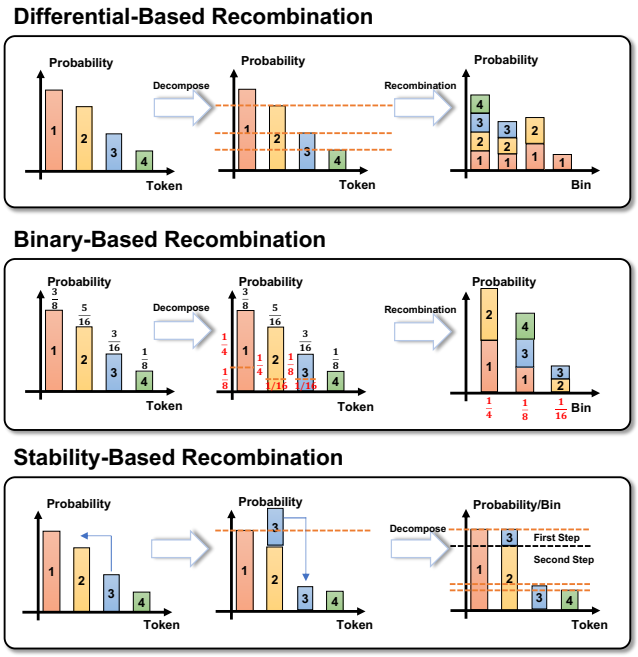

Figure 2: The illustration of typical implementations of probability recombination scheme.

4.2 Probability Recombination Schemes

Before presenting more practical schemes, we show two trivial examples to illustrate the simplicity of creating probability recombination schemes:

- Each token's probability itself acts as a block and is assigned to separate bins.
- Take the two tokens with the highest probabilities. Split the higher one into two blocks, where one block is equal to the lower token's probability. Group the lower token's probability and the equal one block in one bin, with all other tokens' probability and the remaining block in separate bins.

These simple schemes easily satisfy the principles, illustrating how numerous provably secure steganography schemes can be readily constructed when a uniform steganography scheme is available. We will now introduce several non-trivial schemes with intuitive figures shown in Figure 2 and discuss their varied effects in Section 5.

4.2.1 Differential-Based Recombination

The differential-based scheme first rearranges tokens by their probabilities p_i in descending order. Each probability is then decomposed as

$$p_i = \sum_{k=i}^{n} (p_i^k) = \sum_{k=i}^{n} (p_k - p_{k+1}).$$

where $p_{n+1} = 0$ and for each token i, the block $p_i^k = p_k - p_{k+1}$ was assigned to bin G_k. We can verify it as a probability recombination scheme:

1. Block p_i^k is assigned to a bin G_k.
2. All blocks in bin G_k are equal to $p_k - p_{k+1}$.
3. Each bin G_k contains at most one block p_i^k from token i.

This scheme is valid for all distributions and can output the $G = (\mathcal{D}_G, \mathcal{T}_G)$ using Algorithm 3 in $O(n \log n)$ time, where n represents the size of the token vocabulary.

Algorithm 3 Differential-Based Recombination

Input: Distribution $D = [p_i]_n$
Output: $G = (\mathcal{D}_G, \mathcal{T}_G)$

```
D, Indices ← sort(D)                ▷ descending order
Diff ← diff(D)                      ▷ Calculate the difference
for i = 1 to n do
    D_G[i] = Diff[i] × i
    T_G[i] = Indices[:i]
end for
return G = (D_G, T_G)
```

Figure 2 provides a visual demonstration of this scheme. The probabilities are initially represented as bars. Red dashed lines, determined by the heights of the token probabilities, decompose these probabilities. All blocks between any two red lines are then recombined to form a bin.

4.2.2 Binary-Based Recombination

The Binary-Based Recombination scheme first represent the probabilities in binary form with precision m (decided by channel) as

$$p_i = \sum_{k=1}^{m} p_i^k = \sum_{k=1}^{m} b_i^k \cdot 2^{-k}.$$

where $b_i^k \in \{0,1\}$ and the non-zero block $p_i^k = 2^{-k}$ was assigned to bin G_k. We can verify it as a probability recombination scheme:

1. Block p_i^k is assigned to a bin G_k.
2. All blocks in bin G_k are equal to 2^{-k}.
3. Each bin G_k contains at most one block p_i^k from token i.

This scheme is valid for all distributions and outputs G using Algorithm 4 in $O(n)$ time, considering time complexity in terms of n.

Algorithm 4 Binary-Based Recombination

Input: Distribution $D = [p_i]_n$, Precision(fixed) m
Output: $G = (\mathcal{D}_\mathcal{G}, \mathcal{T}_\mathcal{G})$

for j = 1 to m **do**
  for i = 1 to n **do**
    if binary form of p_i contains 2^{-j} **then**
      $\mathcal{T}_\mathcal{G}[j]$.append(i)
    end if
  end for
  $\mathcal{D}_\mathcal{G}[j] = 2^{-j} \cdot |\mathcal{T}_\mathcal{G}[j]|$
end for
return $G = (\mathcal{D}_\mathcal{G}, \mathcal{T}_\mathcal{G})$

4.2.3 Stability-Based Recombination

The Stability-Based Recombination scheme[2] first rearranges tokens by their probabilities p_i in descending order.

If $p_1 > 0.5$, then p_1 is decomposed into blocks as follows

$$p_1 = \sum_{i=1}^{n} p_1^i = (2p_1 - 1) + \sum_{i=2}^{n} p_i.$$

Each of the other tokens acts as a single block. The block $p_1^1 = (2p_1 - 1)$ is assigned to bin G_1, and each block p_1^i is combined with p_i into the bin G_i. We can readily verify that this meets those principles.

If $p_1 \leq 0.5$, we denote k as the minimal number such that $\sum_{i=2}^{k} p_i \geq p_1$.

First Step: We decompose p_k into two blocks: $p_k^1 = \sum_{i=2}^{k} p_i - p_1$ and $p_k^2 = p_k - p_k^1$. We decompose p_1 as $p_1 = \sum_{i=1}^{k-1} p_1^i = p_2 + \sum_{i=3}^{k-1} p_i + p_k^1$. Next, we combine p_1^i with p_i for each $i \in \{3, 4, ..., k-1\}$ and p_1^{k-1} with p_k^1.

Second Step: The remaining blocks p_1^1 (equal to p_2), p_2, p_k^2, p_{k+1}, ..., p_n are then recombined into bins by Differential-Based Recombination. Note that these values may not be in descending order since p_k^2 may be less than p_{k+1}.

The first principle is obvious; for the second and third principles, the bins obtained in the first step clearly meet the requirements, and those from the second step are ensured by the properties of Differential-Based Recombination.

property: An important property to note is that when $p_1 \leq 0.5$, no bin with positive probability contains only one block. In fact, in the first step, all bins contain two blocks, while in the second step, only the first bin with probability $p_{\max} - p_{\text{second}}$ might consist of one block. However, its probability is $p_1^1 - p_2 = 0$.

This scheme is also valid for all distributions and can output $G = (\mathcal{D}_\mathcal{G}, \mathcal{T}_\mathcal{G})$ in $O(n \log n)$ time using Differential-Based Recombination in Algorithm 3.

[2] Stability will be explained in the next section

4.3 Uniform Steganography

The construction of uniform steganography is relatively easier, and some existing constructions can be handily found, such as ones proposed by Ryabko *et al.* [30], which also aligns with the original intent of this proposed framework. Additionally, all provably secure steganography can function as uniform steganography due to their inclusive relationship.

This section only presents a detailed description of one novel construction proposed by us, which is called **Cyclic-shift Uniform Steganography**.

Algorithm 5 Time-Step Encoding: **USE**$(\cdot)$

Input: Distribution $U(n)$, Key K_{US}, Secret Message M
Output: Stegotoken s, Embedded Bit Count c

ptr $\leftarrow \text{PRG}_{K_{\text{US}}}, R \leftarrow \lceil \text{ptr} \cdot n \rceil$
$k \leftarrow \text{floor}(\log_2(n)), m \leftarrow n - 2^k$
bits $\leftarrow M[:k]$, res $\leftarrow M[k]$
idx $\leftarrow$ bits2int(bits)
if idx $< 2^k - m$ **then**
  return $s = T_{(R+idx) \bmod n}, c = k$
else
  return $s = T_{(2idx-(2^k-m)+R+res) \bmod n}, c = k+1$
end if

For an input with a uniform distribution of n tokens $T_1, T_2, ..., T_n$:

$$U(n) = \left[\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right]_n$$

We denote k and m such that $n = 2^k + m$ where $0 \leq m < 2^k$. Thus, we have:

$$2^{-k} \cdot (n - 2m) + 2^{-(k+1)} \cdot 2m = 1.$$

Consider the sequence $B_k = [\text{int2bits}(i,k)]_{i=0}^{2^k-1}$, which represents the binary encoding of numbers from 0 to $2^k - 1$ using k bits. We define a new sequence C_n, consisting of $n(= 2^k + m)$ elements. The first $2^k - m$ elements are the same as those in B_k , and the remaining $2m$ elements are formed by appending '0' or '1' to the last m elements of B_k . For example, $B_2 = [00, 01, 10, \mathbf{11}]$ and $C_5 = [00, 01, 10, \mathbf{110}, \mathbf{111}]$.

We notice that the sequence C_n forms a prefix-free code set, meaning no code in the set is a prefix of any other. And for any bit stream M, there exists an element in C_n that is a prefix of M. We use $C_n(M)$ to denote the index of this element in C_n, for example $C_5('001101') = 1, C_5('1100') = 4$.

At each time step, the sender and receiver first generate a shared pseudo-random number $ptr \in (0, 1]$, which is produced by a *secure PRG* using the shared key K_{US}. By multiplying ptr by n and taking the ceiling value, we obtain R, an integer that can be considered to follow distribution $U(1, n)$. And we define the time-step encoding algorithm as follows:

$$\text{USE}(U(n), K_{\text{US}}, M) = T_{(C_n(M)+R-1) \bmod n},$$

Algorithm 6 Time-Step Decoding: **USD**$(\cdot)$

Input: Distribution $U(n)$, Key K_{US}, Stegotoken $s = T_i$
Output: Secret Message M

ptr $\leftarrow \mathrm{PRG}_{K_{\mathrm{US}}}, R \leftarrow \lceil \mathrm{ptr} \cdot n \rceil$
$k \leftarrow \mathrm{floor}(\log_2(n)), m \leftarrow n - 2^k$
idx $\leftarrow (i - R) \bmod n$ ▷ range from 0 to n-1
if idx $< 2^k - m$ **then**
    return int2bits(idx, k)
else
    $s_1 \leftarrow idx - 2^k + m, s_0 \leftarrow s_1 \bmod 2$ ▷ from 0 to 1
    return $M = \mathrm{int2bits}(\mathrm{floor}(\frac{s_1 - s_0}{2}) + 2^k - t, k) || s_0$
end if

prefix of secret bits	00	01	10	110	111
R = 1	T1	T2	T3	T4	T5
R = 2	T2	T3	T4	T5	T1
R = 3	T3	T4	T5	T1	T2
R = 4	T4	T5	T1	T2	T3
R = 5	T5	T1	T2	T3	T4

Figure 3: An example of $n = 5$

where T_i represent the i-th token and time-step decoding algorithm as follows:

$$\mathrm{USD}(U(n), K_{\mathrm{US}}, T_i) = C_n[i - R + 1 \bmod n].$$

An intuitive illustration of the inverse relationship between the secret message prefix and tokens for $n = 5$ is shown in Figure 3. The corresponding complete time-step algorithms **USE** and **USD** are presented in Algorithm 5 and Algorithm 6.

Correctness The correctness of this uniform steganography scheme can be verified by ensuring the correctness at each time step:

$$\mathrm{USD}(U(n), K_{\mathrm{US}}, \mathrm{USE}(U(n), K_{\mathrm{US}}, M)) = C_n[C_n(M)],$$

which is exactly the prefix of the secret message M, as well as the secret bits embedded at each time step.

Security For the security, we prove that if the *ptr* used at each time step is a truly random number and mutually independent across time steps, then the stegotext matches the original channel distribution $\mathcal{D}$, where all time-step distributions are uniform distributions.

For each time-step distribution $U(n)$[3], the probability of steganographic sampling to any T_i is exactly $\frac{1}{n}$:

$$\begin{aligned} P_{stego}(T_i) &= \sum_{r=1}^{n} P[R = r] \cdot P[C_n(M) = (i - r + 1) \bmod n] \\ &= \frac{1}{n} \cdot \sum_{x=1}^{n} P[C_n(M) = x] = \frac{1}{n}. \end{aligned}$$

The last equality holds since C_n ensures that for any bit stream M, there exists an element in C_n that is a prefix of M. This equation shows that each step of steganographic sampling adheres to the time-step distribution $U(n)$. Additionally, the samplings across different time steps are clearly independent, as the input parameter (*ptr* and the embedded secret message) used in each step are independent. These two properties together guarantee that the stegotext conforms to the channel distribution $\mathcal{D}$. When *ptr* is replaced with pseudo-random numbers generated by a *secure PRG*, it is easy to see that the stegotext and covertext from $\mathcal{D}$ become computationally indistinguishable to any polynomial-time adversary, which satisfies the security definition outlined in Section 2.2.1.

4.4 Discop Revisited

We revisit Discop(base) from the perspective of our framework. Discop can be understood as follows: during each step of sampling, the sender and receiver share a random number *ptr* between 0 and 1. If there exists an n such that this random number *ptr* satisfies $ptr + \frac{1}{2^n} \times k$ landing in different intervals corresponding to tokens I_k, then I_k can be sampled to transmit n-bit information k. Here, n is defined as the largest such value satisfying this condition, and we denote it as $N(ptr) = n$. If n does not exist, we define $N(ptr) = 0$.

Therefore, the intervals corresponding to each token I can be split into several continuous sub-intervals: $A_1, A_2, \ldots, A_l$, such that for all $r \in A_i$, $N(r)$ are the same. Additionally, adjacent sub-intervals within the same token correspond to different n. We denote this as $N(A_i) = N(r)$ for any $r \in A_i$.

Consider the decomposition of the probability of token I into blocks according to $A_1, \ldots, A_l$, note that if $N(A_i) > 0$, then $A_i + \frac{1}{2^n} \times k$ precisely corresponds to other tokens' blocks, so we assign these corresponding 2^n blocks to the same bin. If $N(A_i) = 0$, we assign A_i to its own bin. This can be easily verified as a probability recombination scheme. In this case, each bin contains a number of blocks that is a power of two. Therefore, consider a uniform steganography scheme that adopts the most natural time-step encoding under the distribution $U(2^n)$: first, a random number is used to sample a block as T_0. Then, starting from T_0, the i-th block corresponds to the n-bit binary representation of i. Clearly, *ptr* can be interpreted as being decomposed into the random numbers used in the Bin Sampling and the uniform steganography. Such a scheme is completely equivalent to the Discop(base).

[3] n may vary across different time steps

5 Optimal Schemes Within Our Framework

In this section, we explore an important aspect of our framework: its capacity to guide us in discovering some effective, provably secure steganography schemes for capacity, stability, and time complexity. We demonstrate how to identify these schemes, which are theoretically optimal or nearly so within our framework. Subsequent experimental and theoretical analyses confirm that these schemes perform well in specific metric tests, with some indicators reaching optimal levels compared to other provably secure steganography schemes.

5.1 Capacity

Capacity is one of the most important metrics in steganography. It indicates the amount of secret information that can be carried by a stegotext symbol (token), implying the speed of secret information transmission. Since each sample step involves probabilistic algorithms, we use the expected number of bits that can be hidden under a time-step distribution D, denoted $C_{scheme}(D)$, to represent capacity per token.

In fact, information theory suggests that the upper limit of information that can be transmitted under a distribution, on average, is its entropy. Thus, the upper limit of capacity for any provably secure steganography scheme A is:

$$C_A(D) \leq H(D),^{4}$$

where $H(D) = \sum p_i \log_2\left(\frac{1}{p_i}\right)$ represents the entropy of distribution D.

On the other hand, as the distribution D changes constantly within the channel, the lower bounds of capacity achievable by a scheme are also of interest, indicating how effectively a scheme can utilize entropy. We know from the study [12, 22]:

$$C_{\text{Meteor}}(D) \geq \frac{H(D)}{2} - 0.5, \quad C_{\text{Discop(base)}}(D) \geq H_{\min}(D).$$

These lower bounds are still significantly distant from the upper bounds $H(D)$, suggesting considerable room for theoretical improvement.

We will first explore the near-optimal/optimal schemes of the uniform steganography and probability recombination schemes, focusing on their capacity performance. Subsequently, we will identify the near-optimal provably secure steganography scheme within our framework, then calculate its lower bounds of capacity and evaluate its average capacity in Section 5.4.

5.1.1 Optimal Capacity of Universal Steganography

Similarly, we know that the capacity of uniform steganography applied to a uniform distribution $U(n)$ is limited to

[4]This should refer to the distribution of the stegotoken, but the security of provably secure steganography ensures that these two distributions are nearly identical.

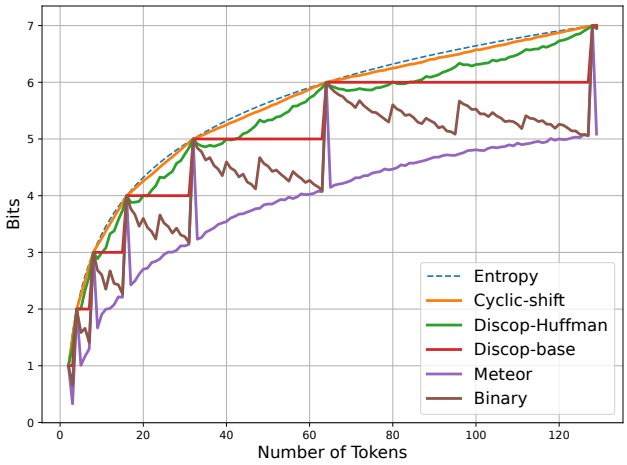

Figure 4: The average capacity of typical implementations of provably secure steganography schemes [12, 22] and uniform steganography by ours and Binary [30], under uniform distribution.

$H(U(n)) = \log_2(n)$. We will next demonstrate that Cyclic-shift Uniform Steganography (Section 4.3) qualifies as a near-optimal scheme, achieving a lower bound of $\log_2(n) - 0.0861$.

For any time step in Cyclic-shift Uniform Steganography, it can hide $|C_n[C_n(M)]|$ bits. Thus in average, its capacity $C(U(n))$ can be calculated:

$$\begin{aligned} C(U(n)) &= \sum_{x \in C_n} P[C_n[C_n(M)] = x] \cdot |x| \\ &= (n-2m) \cdot 2^{-k} \cdot k + 2m \cdot 2^{-(k+1)} \cdot (k+1) \\ &= k + m \cdot 2^{-k}. \end{aligned}$$

Then,

$$\begin{aligned} H(U(n)) - C(U(n)) &= \log_2(n) - k + m \cdot 2^{-k} \\ &= \log_2(1 + \frac{m}{2^k}) - \frac{m}{2^k} = \log_2(1+x) - x. \end{aligned}$$

where $x = \frac{m}{2^k} < 1$. And $0 \leq \log_2(1+x) - x < 0.0861$, which can be easily calculated. Thus:

$$H(U(n)) \geq C(U(n)) > H(U(n)) - 0.0861. \tag{2}$$

As a comparison, Figure 4 tests the actual capacities of various provably secure steganography schemes [12, 22] and uniform steganography (ours and [30]) under uniform distributions. It is easy to see that Cyclic-shift Uniform Steganography is near-optimal and surpasses all current schemes.

5.1.2 Optimal Scheme of Probability Recombination

Regarding the impact of Probability Recombination Module on the capacity, we have obtained quite good theoretical results, summarized as follows: If there exists a uniform

steganography that its capacity achieves $H(U(n)) = \log_2(n)$, we can prove that Differential-Based Recombination has the optimal capacity for all distributions D:

Lemma 1 (Optimal Scheme)**.** *Assume $p_1 \geq p_2 \geq \cdots \geq p_n \geq 0$, satisfying $\sum_{i=1}^{n} p_i = 1$. Given any recombination scheme $\mathcal{G} = (\mathcal{D}_{\mathcal{G}} = [|T_i| \cdot q_i]_n, \mathcal{T}_{\mathcal{G}} = \{T_1, \cdots, T_m\})$ The capacity of this scheme is*

$$C' = \sum_{j=1}^{m} q_j |T_j| \log_2 |T_j|.$$

Then we have the following inequality:

$$C' \leq \sum_{i=1}^{n} (p_i - p_{i+1}) i \log_2 i,$$

where $p_{n+1} := 0$.

The right-hand side of the final inequality represents the capacity of the Differential-Based Recombination. And we can calculate the lower bound of this optimal capacity:

Lemma 2 (Lower Bound of Optimal Capacity)**.** *Assume $p_1 \geq p_2 \geq \cdots \geq p_n \geq 0$, satisfying $\sum_{i=1}^{n} p_i = 1$. Denote:*

$$C = \sum_{i=1}^{n} (p_i - p_{i+1}) i \log_2 i, \quad H = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}, \tag{3}$$

where $0 \cdot \log_2 0 := 0, p_{n+1} := 0$. Then the following inequalities hold:

$$H - \log_2(1 + H \cdot \ln 2) \leq C \leq H. \tag{4}$$

We prove these 2 lemmas in the Appendix A and B. From Equation 2 and Lemma 2, it is evident that provably secure steganography derived from our framework, utilizing Differential-Based Recombination and Cyclic-shift Uniform Steganography, can achieve a lower bound of $H - \log_2(1 + H \cdot \ln 2) - 0.0861$, which classifies it as a near-optimal scheme in our framework. This lower bound surpasses that of Meteor[5] under all conditions, as illustrated in Figure 5. Thus, our scheme currently has the best lower bound. Furthermore, this clearly represents an asymptotically optimal lower bound.

5.2 Stability

Steganography, as a form of communication, should ensure that the amount of secret information transmitted at each time step is greater than zero, as embedding information consumes computational resources, and each step should be as productive as possible to minimize unnecessary overhead. However, due to the inherently probabilistic nature of steganographic

[5]Since Discop(Huffman) does not provide a precise lower bound, this is the best-known lower bound.

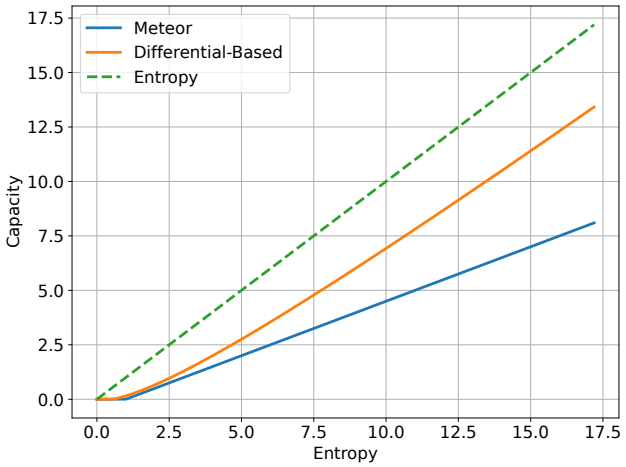

Figure 5: The lower bound of capacity of ours and Meteor.

encoding, even schemes with high capacity may still exhibit a significant probability of embedding zero bits in certain cases. For instance, a scheme might embed 0 bits with a 99% probability and 10,000 bits with a 1% probability. While this results in a high capacity (an average concept), the scheme is nearly impractical in real-world scenarios. To address this issue, we introduce the concept of "stability" to describe the property of consistently embedding a non-zero amount of information at each time step. This concept complements capacity and provides a more comprehensive characterization of the performance of steganographic schemes, particularly in scenarios where the cost of operations is high. To achieve stability, a necessary condition related to the time-step distribution is that its minimum entropy $\log_2(\frac{1}{p_{max}})$ should be at least 1 bit ($\Leftrightarrow p_{max} \leq 0.5$). This ensures that each token's information content $\log_2(\frac{1}{p_i})$ is at least 1 bit. We consider a scheme **stable** if it can guarantee the embedding of at least 1 bit for all distributions where the minimum entropy is greater than or equal to 1 bit.

Currently, only the Discop(base) [12] scheme is stable. In contrast, the scheme with higher capacity, like Meteor [22], Discop(Huffman) [12] and the scheme with differential-based recombination, cannot consistently achieve this (see Appendix C for examples).

Note that for Cyclic-shift Uniform Steganography, embedding 0 bits is equivalent to the Bin Sampling sample a bin with only one block. Therefore, if we design a probability recombination scheme such that no such bin exists when the minimum entropy is greater than 1 bit, stable scheme can be achieved. And since the **property** in Section 4.2.3, the combination of Stability-Based Recombination scheme and Cyclic-shift Uniform Steganography forms a stable steganography scheme. And comparing to Discop(base) in Section 5.4, our scheme has higher capacity.

Table 1: The main results of our proposed framework. C means average Capacity (bits per token). H means the average Entropy (bits per token). C/H means the utilization of entropy. NR means the No-payload Rate (the frequency of embedding 0 bits). T means average Time (second per token). Scenario A is conducted by Llama3 [34] model based on the history "Tell me something about Steganography". Scenario B is conducted by Llama3 model based on the history "1+1=?". Scenario C and D are conducted on uniform distribution channel with 10 and 1,000,000 tokens, respectively.

Scenario A

	C	H	C/H	NR	T
Meteor	0.372	0.697	0.534	0.820	**0.0008**
Meteor-reorder	0.438	0.709	0.618	0.798	2.116
Discop-base	0.343	0.700	0.489	0.687	0.0200
Discop-Huffman	**0.518**	0.691	**0.749**	0.691	0.0461
Binary-based	0.224	0.702	0.320	0.847	0.0033
Stability-based	0.366	0.701	0.523	**0.686**	0.0012
Differential-based	0.410	0.693	0.592	0.729	0.0025
random-sample	0.000	0.698	0.000	1.000	0.0003

Scenario B

	C	H	C/H	NR	T
Meteor	0.283	0.875	0.323	0.764	**0.0009**
Meteor-reorder	0.661	0.872	0.758	0.660	3.3286
Discop-base	0.631	0.873	0.723	**0.586**	0.0198
Discop-Huffman	**0.702**	0.873	**0.804**	0.592	0.0701
Binary-based	0.490	0.872	0.562	0.723	0.0031
Stability-based	0.646	0.872	0.740	**0.586**	0.0012
Differential-based	0.670	0.875	0.766	0.610	0.0024
random-sample	0.000	0.873	0.000	1.000	0.0009

Scenario C

	C	H	C/H	NR	T
Meteor	1.804	3.322	0.543	0.100	0.0006
Meteor-reorder	1.804	3.322	0.543	0.100	0.0010
Discop-base	3.000	3.322	0.903	**0.000**	0.0006
Discop-Huffman	3.001	3.322	0.904	**0.000**	**0.0005**
Binary-based	**3.247**	3.322	**0.978**	**0.000**	0.0030
Stability-based	**3.247**	3.322	**0.978**	**0.000**	0.0009
Differential-based	**3.247**	3.322	**0.978**	**0.000**	0.0007
random-sample	0.000	3.322	0.000	1.000	0.0001

Scenario D

	C	H	C/H	NR	T
Meteor	17.91	19.93	0.899	**0.000**	**0.0009**
Meteor-reorder					>60
Discop-base					>60
Discop-Huffman	19.77	19.93	0.992	**0.000**	0.3649
Binary-based	**19.90**	19.93	**0.998**	**0.000**	0.0061
Stability-based	**19.90**	19.93	**0.998**	**0.000**	0.0011
Differential-based	**19.90**	19.93	**0.998**	**0.000**	0.0056
random-sample	0.00	19.93	0.000	1.000	0.0003

5.3 Time Complexity

In steganography, the time complexity of time-step encoding algorithm is a significant point of discussion (decoding algorithms generally have a similar time complexity). Among the various factors, the most important variable is the size of the token vocabulary. Therefore, we ignore the total input length and primarily focus on the time complexity with respect to this variable. For instance, in text channels, the number of tokens used to model the channel has increased from 50k in the GPT-2 [28] era to 128k today for Llama3 [34], which has made some schemes with higher time complexities impractical for real-world applications [10].

For a given distribution, the time complexity of random sampling according to probabilities is $O(n)$, where n is the size of the token vocabulary. Thus, $O(n)$ should also be the optimal of complexity for steganographic sampling. Within our framework, we note that the time complexity of Cyclic-shift Uniform Steganography does not exceed $O(n)$. Therefore, when combined with the Binary-Based Recombination scheme with a time complexity of $O(n)$ and a Bin Sampling process that also does not exceed $O(n)$, we can achieve a provably secure steganography scheme with a time complexity of $O(n)$.

5.4 Experiments

Setup We have compared the performance of several schemes constructed based on our framework with other schemes on the most commonly used channels: text channels and uniform distribution channels. We focused primarily on metrics such as average capacity, average time, and stability. Specifically, for experiments in text channels, we utilized the large language model Llama-3-8B-Instruct [6] and various prompts as historical information to construct different channels. The first channel with prompt "Tell me something about steganography" is designed for long texts, while the second channel with prompt "1+1=?" is suited for short texts. Examples of constructed text steganography are presented in the appendix D. The maximum channel length was limited to 128. For uniform distribution channels, we selected smaller scales

[6] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

of 10 and larger scales of 1,000,000 for comparison. Unlike low-entropy text channels, uniform distribution channels simulate highly random environments, providing a more accurate reflection of the steganography scheme's performance in high-randomness settings. We implemented tests for the Meteor, Meteor(reorder), Discop(base), and Discop(Huffman) based on their source code repositories[7][8]. Notably, Discop(base) and Discop(Huffman) were built and optimized using Cython, while the remaining code was implemented only using the PyTorch framework and Python programming language. All of our provably secure steganography schemes are based on the corresponding probability recombination scheme with Cyclic-shift Uniform Steganography. The precision for all schemes is set to 52 by default.

All our experiments were conducted on 1 $\times$ NVIDIA A5000 GPU (32GB RAM) and 24 $\times$ Intel Xeon w5-3423 CPUs. Additionally, we implemented the PRG functionality using *HMAC_DRBG*, same as in [22]. Main experimental results are presented in Table 1.

Security To further validate the practical security of these schemes, we conducted classic steganalysis experiments [41] on our samples within text channels (Scenario A, B) and statistical analysis on samples in uniform channels (Scenario C). The steganalysis is a task to detect stegotexts, and the classification accuracy is our adopted metric. The statistical analysis adopts Kullback-Leibler divergence (KLD) as the metric, whose calculation can be found in Section 2.1. Table 4 in Appendix E reports a steganalysis accuracy score approaching 50% and also reports a very low KLD. The experimental results demonstrate that our schemes can maintain practical security across these channels.

Capacity Regarding capacity, we focus on the Differential-based scheme. We observed on the 2 text channels constructed for the experiments (which have lower average entropy) a distinct trend: Discop(Huffman) $>$ Differential-based $\approx$ Meteor(reorder) $>$ others. However, Discop(Huffman) and Meteor(reorder) both exhibit significantly higher time consumption, even Discop(Huffman) is implemented with Cython optimization. In experiments on uniform channels (with higher entropy), our method performs markedly better, as suggested by the theoretical analysis in Equation 2. Similarly, Figure 4 also demonstrates the same effect.

We emphasize that due to the variability of distributions and channels, the capacity and conclusions of different methods vary across channel types. We only present very few results, and we also note that Discop(Huffman) indeed remains the optimal method across nearly all experimental channels.

[7] https://meteorfrom.space

[8] https://github.com/comydream/Discop

Stability In the results, we also have measured the frequency of embedding 0 bits across different channels, which corresponds to the metric NR in the Table 1. We observed that schemes which are stable, namely Discop(base) and Stability-based, exhibit significantly lower frequencies. In contrast, the unstable schemes, even those with high capacity such as Discop(Huffman), show slightly weaker performance on this metric. And compared to Discop(base) in Table 1, our Stability-based scheme appears to have higher capacity.

Time The times reported in Table 1 refer to the average time taken to sample a token for a given time-step distribution D, excluding the time spent on calculating the time-step distribution. We observe that Meteor remains the best-performing method, nearly equivalent to random sampling. On the other hand, despite optimizations using Cython, the Discop(Huffman) scheme shows significantly poorer performance, likely due to the inherent time required to construct a Huffman tree. Meteor(reorder), meanwhile, exhibits notably worse efficiency, a disparity that becomes more pronounced as the number of tokens n in the channel increases. Regarding our schemes, generally, they perform very well in terms of time. Although the Binary-based scheme is $O(n)$, it does not necessarily outperform the $O(n \log n)$ schemes. This could be due to better optimization of sorting algorithms internally and the impact of larger constants — notably precision — affecting the results.

We also note that different methods exhibit diverse effects across various channels. This variability underscores the necessity of our framework, which can provide more options when dealing with unpredictable distributions and channels. Therefore, further research is warranted.

6 Conclusion

In this paper, we propose a framework for designing new provably secure steganography schemes by decomposing the whole design process into three sub-processes that can be relatively easily achieved, namely Probability Recombination Module, Bin Sampling and Uniform Steganography Module. Under the guidance of this framework, we construct a series of provably secure steganography schemes that exhibit theoretical advantages across various metrics. Experiments have also confirmed that our schemes perform well in practical applications across these metrics. This framework not only aids in developing schemes tailored for specific metrics but also advances the field by making the design of provably secure steganography schemes more convenient.

Acknowledgments

This work was supported by the Natural Science Foundation of China under Grants 62262002 and U2336208.

7 Ethics considerations

We recognize that the steganography may be used to evade legitimate scrutiny and provide convenience for some activities of malicious purposes. However, since there are already some existing provably secure steganography constructions, our work will not make this ethical concern (potential malicious usages of steganography) much worse. Besides, our work provides a novel framework and a mathematical analytical perspective for many (current and future) provably secure steganography constructions, which we believe is necessary to ensure that we have at our disposal the tools needed to prevent steganography from being used for malicious purposes.

Apart from the aforementioned ethical concern, the authors foresee no other ethical concerns with the work presented in this paper.

8 Open science

Our work is in compliance with the Open Science Policy. The source code for our work is available at https://zenodo.org/records/14737116. This repository contains the full implementation of the schemes discussed in the paper, along with scripts and documentation to assist in running the work.

References

[1] R.J. Anderson and F.A.P. Petitcolas. On the limits of steganography. *IEEE Journal on Selected Areas in Communications*, 16(4):474–481, May 1998.

[2] Michael Backes and Christian Cachin. Public-Key Steganography with Active Attacks. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, and Joe Kilian, editors, *Theory of Cryptography*, volume 3378, pages 210–226. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[4] Sebastian Berndt and Maciej Liśkiewicz. Provable Secure Universal Steganography of Optimal Rate: Provably Secure Steganography does not Necessarily Imply One-Way Functions. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, pages 81–92, Vigo Galicia Spain, June 2016. ACM.

[5] Sebastian Berndt and Maciej Liśkiewicz. On the Gold Standard for Security of Universal Steganography. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018*, volume 10820, pages 29–60. Springer International Publishing, Cham, 2018.

[6] Nikita Borisov, Ian Goldberg, and Eric Brewer. Off-the-record communication, or, why not to use pgp. In *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 77–84, 2004.

[7] Christian Cachin. An information-theoretic model for steganography. In *International Workshop on Information Hiding*, pages 306–318. Springer, 1998.

[8] Kejiang Chen, Hang Zhou, Yaofei Wang, Menghan Li, Weiming Zhang, and Nenghai Yu. Cover reproducible steganography via deep generative models. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[9] Falcon Dai and Zheng Cai. Towards near-imperceptible steganographic text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4303–4308, 2019.

[10] Christian Schroeder de Witt, Samuel Sokota, J Zico Kolter, Jakob Nicolaus Foerster, and Martin Strohmeier. Perfectly secure steganography using minimum entropy coupling. In *The Eleventh International Conference on Learning Representations*, 2022.

[11] Mario Di Raimondo, Rosario Gennaro, and Hugo Krawczyk. Secure off-the-record messaging. In *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, pages 81–89, 2005.

[12] Jinyang Ding, Kejiang Chen, Yaofei Wang, Na Zhao, Weiming Zhang, and Nenghai Yu. Discop: Provably Secure Steganography in Practice Based on "Distribution Copies". In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 2238–2255. IEEE Computer Society, 2023.

[13] Roger Dingledine, Nick Mathewson, Paul F Syverson, et al. Tor: The second-generation onion router. In *USENIX security symposium*, volume 4, pages 303–320, 2004.

[14] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, 2022.

[15] Tina Fang, Martin Jaggi, and Katerina Argyraki. Generating steganographic text with lstms. In *Proceedings of ACL 2017, Student Research Workshop*, pages 100–106, 2017.

[16] John Bellamy Foster and Robert McChesney. Surveillance capitalism. *Monthly review*, 66(3):1–31, 2014.

[17] Phillipa Gill, Masashi Crete-Nishihata, Jakub Dalek, Sharon Goldberg, Adam Senft, and Greg Wiseman. Characterizing web censorship worldwide: Another look at the opennet initiative data. *ACM Transactions on the Web (TWEB)*, 9(1):1–29, 2015.

[18] Nicholas J Hopper. *Toward a Theory of Steganography*.

[19] Nicholas J Hopper and John Langford. Provably Secure Steganography.

[20] Yuting Hu, Yihua Huang, Zhongliang Yang, and Yongfeng Huang. Detection of heterogeneous parallel steganography for low bit-rate voip speech streams. *Neurocomputing*, 419:70–79, 2021.

[21] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[22] Gabriel Kaptchuk, Tushar M Jois, Matthew Green, and Aviel D Rubin. Meteor: Cryptographically secure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1529–1548, 2021.

[23] Arash Habibi Lashkari, Gerard Draper Gil, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. Characterization of tor traffic using time based features. In *International Conference on Information Systems Security and Privacy*, volume 2, pages 253–262. SciTePress, 2017.

[24] Zhen Ling, Junzhou Luo, Kui Wu, Wei Yu, and Xinwen Fu. Torward: Discovery, blocking, and traceback of malicious traffic over tor. *IEEE Transactions on Information Forensics and Security*, 10(12):2515–2530, 2015.

[25] David Lyon. *Surveillance society*. McGraw-Hill Education (UK), 2001.

[26] Xuying Meng, Chungang Lin, Yequan Wang, and Yujun Zhang. Netgpt: Generative pretrained transformer for network traffic. *arXiv preprint arXiv:2304.09513*, 2023.

[27] Yuang Qi, Kejiang Chen, Kai Zeng, Weiming Zhang, and Nenghai Yu. Provably secure disambiguating neural linguistic steganography. *arXiv preprint arXiv:2403.17524*, 2024.

[28] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[29] Michael G Reed, Paul F Syverson, and David M Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected areas in Communications*, 16(4):482–494, 1998.

[30] Boris Ryabko and Daniil Ryabko. Information-theoretic approach to steganographic systems. In *2007 IEEE International Symposium on Information Theory*, pages 2461–2464. IEEE, 2007.

[31] Ferry Astika Saputra, Isbat Uzzin Nadhori, and Balighani Fathul Barry. Detecting and blocking onion router traffic using deep packet inspection. In *2016 International Electronics Symposium (IES)*, pages 283–288. IEEE, 2016.

[32] Jiaming Shen, Heng Ji, and Jiawei Han. Near-imperceptible neural linguistic steganography via self-adjusting arithmetic coding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 303–313, 2020.

[33] Ram Sundara Raman, Prerana Shenoy, Katharina Kohls, and Roya Ensafi. Censored planet: An internet-wide, longitudinal censorship observatory. In *proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 49–66, 2020.

[34] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[35] Tri Van Le. Efficient provably secure public key steganography. *Cryptology ePrint Archive*, 2003.

[36] Peter Wayner. Mimic functions. *Cryptologia*, 16(3):193–214, 1992.

[37] Hanzhou Wu, Biao Yi, Feng Ding, Guorui Feng, and Xinpeng Zhang. Linguistic Steganalysis With Graph Neural Networks. *IEEE Signal Processing Letters*, 28:558–562, 2021.

[38] Jinshuai Yang, Zhongliang Yang, Siyu Zhang, Haoqin Tu, and Yongfeng Huang. SeSy: Linguistic Steganalysis Framework Integrating Semantic and Syntactic Features. *IEEE Signal Processing Letters*, 29:31–35, 2022.

[39] Jinshuai Yang, Zhongliang Yang, Jiajun Zou, Haoqin Tu, and Yongfeng Huang. Linguistic Steganalysis Toward Social Network. *IEEE Transactions on Information Forensics And Security*, 18, 2023.

[40] Zhong-Liang Yang, Xiao-Qing Guo, Zi-Ming Chen, Yong-Feng Huang, and Yu-Jin Zhang. RNN-Stega: Linguistic Steganography Based on Recurrent Neural Networks. *IEEE Transactions on Information Forensics and Security*, 14(5):1280–1295, 2019.

[41] Zhongliang Yang, Yongfeng Huang, and Yu-Jin Zhang. TS-CSW: Text steganalysis and hidden capacity estimation based on convolutional sliding windows. *Multimedia Tools and Applications*, 79(25-26):18293–18316, July 2020.

[42] Siyu Zhang, Zhongliang Yang, Jinshuai Yang, and Yongfeng Huang. Provably secure generative linguistic steganography. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3046–3055, 2021.

[43] Zachary Ziegler, Yuntian Deng, and Alexander M Rush. Neural linguistic steganography. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1210–1215, 2019.

[44] Jiajun Zou, Zhongliang Yang, Siyu Zhang, Sadaqat Ur Rehman, and Yongfeng Huang. High-Performance Linguistic Steganalysis, Capacity Estimation and Steganographic Positioning. In Xianfeng Zhao, Yun-Qing Shi, Alessandro Piva, and Hyoung Joong Kim, editors, *Digital Forensics and Watermarking*, volume 12617, pages 80–93. Springer International Publishing, Cham, 2021.

A Lower Bound of Capacity

Theorem 1. *Assume* $p_1 \geq p_2 \geq \cdots \geq p_n \geq 0$*, satisfying* $\sum_{i=1}^{n} p_i = 1$*. Denote:*

$$C = \sum_{i=1}^{n} (p_i - p_{i+1}) i \log_2 i, \quad H = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}, \tag{5}$$

where $0 \cdot \log_2 0 := 0$, $p_{n+1} := 0$*. Then the following inequalities hold:*

$$H - \log_2(1 + H \cdot \ln 2) \leq C \leq H. \tag{6}$$

Proof. We first claim that for any non-negative sequences x_i, y_i,

$$\sum_{i=1}^{n} x_i \log_2 \frac{y_i}{x_i} \leq \left(\sum_{i=1}^{n} x_i \right) \log_2 \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i}. \tag{7}$$

This follows from Jensen's inequality and the fact that $-\log_2$ is a convex function. Therefore, we have

$$\sum_{i=j}^{n} (p_i - p_{i+1}) \log_2 i \leq p_j \log_2 \frac{\sum_{i=j}^{n} (p_i - p_{i+1}) i}{p_j} \leq p_j \log_2 \frac{1}{p_j}.$$

Here we use the fact that

$$\sum_{i=j}^{n} (p_i - p_{i+1}) i = j p_j + \sum_{i=j+1}^{n} p_i \leq \sum_{i=1}^{n} p_i \leq 1.$$

And then it shows that

$$C = \sum_{i=1}^{n} (p_i - p_{i+1}) i \log_2 i = \sum_{j=1}^{n} \sum_{i=j}^{n} (p_i - p_{i+1}) \log_2 i$$
$$\leq \sum_{j=1}^{n} p_j \log_2 \frac{1}{p_j} = H.$$

Hence the upper bound of (6) is proved.

For the lower bound of (6), it's sufficient to prove

$$C - \alpha H \geq \frac{\alpha}{\ln 2} + \log_2(1 - \alpha), \quad \forall \alpha \in (0, 1), \tag{8}$$

and the lower bound in (6) follows by taking $\alpha = \frac{H\cdot\ln 2}{1+H\cdot\ln 2}$ in the above inequality.

Define a function $p:[0,n)\to\mathbb{R}$, $p(x)=p_i, \forall x\in[i-1,i)$. It's easy to verify that $p(x)$ is mesurable, and then all the following integrals are Lebesgue Integrable. Further more, the following equations also hold:

$$\int_0^n p(x)\mathrm{d}x = \sum_{i=1}^n p_i = 1, \tag{9}$$

$$H = -\int_0^n p(x)\log_2 p(x)\mathrm{d}x. \tag{10}$$

Meanwhile we have

$$C = \sum_{i=1}^n (p_i - p_{i+1}) i\log_2 i = \sum_{i=1}^n p_i(i\log_2 i - (i-1)\log_2(i-1)). \tag{11}$$

Note that $\log_2 x$ is absolutely integrable in $[0,n]$, we have the Newton-Leibniz formula

$$b\log_2 b - a\log_2 a = \int_a^b \left(\frac{1}{\ln 2}+\log_2 x\right)\mathrm{d}x, \quad \forall 0\le a\le b.$$

Hence (11) is equivalent to the following equation:

$$\begin{aligned} C &= \sum_{i=1}^n p_i \int_{i-1}^i \left(\frac{1}{\ln 2}+\log_2 x\right)\mathrm{d}x \\ &= \sum_{i=1}^n \int_{i-1}^i p(x)\left(\frac{1}{\ln 2}+\log_2 x\right)\mathrm{d}x \\ &= \int_0^n p(x)\left(\frac{1}{\ln 2}+\log_2 x\right)\mathrm{d}x \\ &= \frac{1}{\ln 2} + \int_0^n p(x)\log_2 x\mathrm{d}x. \end{aligned} \tag{12}$$

Here the last equality holds from (9). Therefore, combining eqs. (10) and (12) gives

$$C - \alpha H = \frac{1}{\ln 2} + \int_0^n p(x)\log_2(xp(x)^\alpha)\mathrm{d}x. \tag{13}$$

On the other hand, it's obvious that $\log_2 a = -\frac{1}{\ln 2}\int_a^1 \frac{1}{t}\mathrm{d}t$ holds for any $0<a\le 1$.
Substitute it into (13) gives

$$\begin{aligned} C - \alpha H - \frac{1}{\ln 2} &\ge \int_0^n p(x)\log_2(\min\{xp(x)^\alpha, 1\})\mathrm{d}x \\ &= -\frac{1}{\ln 2}\int_0^n p(x)\left(\int_{\min\{xp(x)^\alpha,1\}}^1 \frac{1}{t}\mathrm{d}t\right)\mathrm{d}x \\ &= -\frac{1}{\ln 2}\int_0^1 \frac{1}{t}\int_{\{x|xp(x)^\alpha\le t\}} p(x)\mathrm{d}x\mathrm{d}t. \end{aligned} \tag{14}$$

Denote

$$\begin{aligned} S(t) &= \int_0^n \min\left\{p(x), \left(\frac{t}{x}\right)^{1/\alpha}\right\}\mathrm{d}x \\ &= \int_{\{x|xp(x)^\alpha\le t\}} p(x)\mathrm{d}x + \int_{\{x|xp(x)^\alpha> t\}} \left(\frac{t}{x}\right)^{1/\alpha}\mathrm{d}x. \end{aligned} \tag{15}$$

Since $p_i\le 1$, we have $p(x)\le 1$, and then

$$\begin{aligned} S(t) &\le \int_0^\infty \min\left\{1, \left(\frac{t}{x}\right)^{1/\alpha}\right\}\mathrm{d}x \\ &= \int_0^t 1\mathrm{d}x + \int_t^\infty \left(\frac{t}{x}\right)^{1/\alpha}\mathrm{d}x \\ &= t + \frac{\alpha}{\alpha-1}t^{1/\alpha}x^{1-1/\alpha}\Big|_t^\infty \\ &= \frac{t}{1-\alpha}. \end{aligned}$$

On the other hand, we have $S(t)\le\int_0^n p(x)\mathrm{d}x = 1$, thus we can tell

$$S(t) \le \min\left\{\frac{t}{1-\alpha}, 1\right\}. \tag{16}$$

Combining eqs. (15) and (16) we get

$$\int_{\{x|xp(x)^\alpha\le t\}} p(x)\mathrm{d}x \le \min\left\{\frac{t}{1-\alpha},1\right\} - \int_{\{x|xp(x)^\alpha>t\}} \left(\frac{t}{x}\right)^{1/\alpha}\mathrm{d}x.$$

Substitute it into (14), it shows

$$\begin{aligned} &C - \alpha H - \frac{1}{\ln 2} \\ &\quad\ge -\frac{1}{\ln 2}\int_0^1 \frac{1}{t}\min\left\{\frac{t}{1-\alpha},1\right\}\mathrm{d}t \\ &\quad + \frac{1}{\ln 2}\int_0^1 \frac{1}{t}\int_{\{x|xp(x)^\alpha>t\}} \left(\frac{t}{x}\right)^{1/\alpha}\mathrm{d}x\mathrm{d}t. \end{aligned} \tag{17}$$

Meanwhile we have

$$\begin{aligned} \int_0^1 \frac{1}{t}\min\left\{\frac{t}{1-\alpha},1\right\}\mathrm{d}t &= \int_0^{1-\alpha}\frac{1}{1-\alpha}\mathrm{d}t + \int_{1-\alpha}^1 \frac{1}{t}\mathrm{d}t \\ &= 1 - \ln(1-\alpha), \end{aligned} \tag{18}$$

$$\begin{aligned} \int_0^1 \frac{1}{t}\int_{\{x|xp(x)^\alpha>t\}} \left(\frac{t}{x}\right)^{1/\alpha}\mathrm{d}x\mathrm{d}t &= \int_0^n \int_0^{xp(x)^\alpha} \frac{t^{1/\alpha-1}}{x^{1/\alpha}}\mathrm{d}t\mathrm{d}x \\ &= \int_0^n \frac{\alpha x^{1/\alpha}p(x)}{x^{1/\alpha}}\mathrm{d}x \\ &= \alpha\int_0^n p(x)\mathrm{d}x = \alpha. \end{aligned} \tag{19}$$

Combining eqs. (17) to (19) we obtain (8):

$$C - \alpha H \ge \frac{1-(1-\ln(1-\alpha))+\alpha}{\ln 2} = \frac{\alpha}{\ln 2} + \log_2(1-\alpha).$$

By taking $\alpha = \frac{H\cdot\ln 2}{1+H\cdot\ln 2}$ in (8), we finish the proof. □

B Optimal of Capacity

Theorem 2. *Assume $p_1\ge p_2\ge\cdots\ge p_n\ge 0$, satisfying $\sum_{i=1}^n p_i = 1$. Given any recombination scheme: $T_1,\cdots,T_m$,*

where each T_i is a subset of $\{1,\cdots,n\}$, and the probability of every block in T_i is q_i. Define the capacity of this scheme by

$$C' = \sum_{j=1}^{m} q_j |T_j| \log_2 |T_j|.$$

Then we have the following inequality:

$$C' \le \sum_{i=1}^{n} (p_i - p_{i+1}) i \log_2 i,$$

where $p_{n+1} := 0$.

Proof. By definition, we have

$$p_i = \sum_{i \in T_j} q_j. \tag{20}$$

Since $|T_j|$ takes value from $\{1,2,\cdots,n\}$, we denote $r_k = \sum_{|T_j|=k} q_j$, and then

$$C' = \sum_{k=1}^{n} r_k k \log_2 k.$$

Let $f(x) = \begin{cases} x\log_2 x, & x > 0; \\ 0, & x = 0. \end{cases}$, it's obvious that $f(x)$ is convex. Then we have

$$\begin{aligned} C' &= \sum_{k=1}^{n} r_k f(k) = \sum_{k=1}^{n}\sum_{j=k}^{n} r_j (f(k) - f(k-1)) \\ &= \sum_{k=1}^{n}\sum_{j=k}^{n}\sum_{i=j}^{n} r_i [f(k) - 2f(k-1) + f(k-2)] \\ &= \sum_{k=1}^{n} \left(\sum_{j=k}^{n} (j-k+1) r_j \right) [f(k) - 2f(k-1) + f(k-2)]. \end{aligned} \tag{21}$$

Here we denote $f(-1) := 0$. When $k \ge 2$, by the convexity of f, we have $f(k) - 2f(k-1) + f(k-2) \ge 0$; when $k = 1$, we have $f(k) - 2f(k-1) + f(k-2) = 0$. Hence we have

$$f(k) - 2f(k-1) + f(k-2) \ge 0, \quad \forall k \ge 1 \tag{22}$$

In the following, we prove

$$\sum_{j=k}^{n} (j-k+1) r_j \le \sum_{i=k}^{n} p_i. \tag{23}$$

Firstly, we have

$$\begin{aligned} \sum_{j=k}^{n} (j-k+1) r_j &= \sum_{j=1}^{n} \max\{j-k+1, 0\} r_j \\ &= \sum_{j=1}^{n} j r_j - \sum_{j=1}^{n} \min\{k-1, j\} r_j. \end{aligned} \tag{24}$$

Recall that $r_j = \sum_{|T_l|=j} q_l$, we know

$$\sum_{j=1}^{n} j r_j = \sum_{l=1}^{m} q_l |T_l| = \sum_{i=1}^{n} \sum_{i \in T_l} q_l = \sum_{i=1}^{n} p_i, \tag{25}$$

where the last equality is due to (20). And

$$\sum_{j=1}^{n} \min\{k-1, j\} r_j = \sum_{l=1}^{m} q_l \min\{k-1, |T_l|\}.$$

Denote $[k] := \{1,2,\cdots,k\}, \forall k \ge 1$, $[0] := \emptyset$. Since $|T_l \cap [k-1]| \le \min\{k-1, |T_l|\}$, we get

$$\begin{aligned} \sum_{j=1}^{n} \min\{k-1, j\} r_j &\ge \sum_{l=1}^{m} q_l |T_l \cap [k-1]| \\ &= \sum_{i=1}^{n} \sum_{i \in T_l \cap [k-1]} q_l \\ &= \sum_{i=1}^{k-1} p_i, \end{aligned} \tag{26}$$

where the last equality is due to (20). Combining Equations (24) to (26) gives (23).

Substitute (23) into (21) (notice that (22) holds), we get

$$\begin{aligned} C' &= \sum_{k=1}^{n} \left(\sum_{j=k}^{n} (j-k+1) r_j \right) [f(k) - 2f(k-1) + f(k-2)] \\ &\le \sum_{k=1}^{n} \left(\sum_{i=k}^{n} p_i \right) [f(k) - 2f(k-1) + f(k-2)] \\ &= \sum_{k=1}^{n} p_k (f(k) - f(k-1)) \\ &= \sum_{k=1}^{n} (p_k - p_{k-1}) f(k). \end{aligned}$$

This completes the proof. □

C Stability of Meteor and Discop

Here, we demonstrate with illustrations why we say the Discop(Huffman) (Figure 6) and Meteor (Figure 7) are not stable. In these shown situaiton, the $P_{max} \le 0.5$ but Discop(Huffman) and Meteor still may not embed any bits.

D Demo of Steganography Text

We show some demo stegotext produced by existing provably secure generative steganography and ours constructions, under different history, see Table 2 and 3.

Table 2: Resulting stegotext based on the history: “Tell me something about Steganography”

	stegotext	hidden bits
Meteor	Steganography! It’s a fascinating and less well-known branch of cryptography...	01100110...
Meteor-reorder	Steganography! It’s a fascinating field that combines art and science...	01100110...
Discop-base	Steganography! It’s a fascinating field that combines cryptography and computer science...	01100110...
Discop-Huffman	Steganography!\n\nSteganography is the practice of hiding secret information...	01100110...
Binary-based	Steganography! It’s a fascinating and ancient concept that has been used to conceal messages...	01100110...
Stability-based	Steganography!\n\nSteganography is the art of hiding secret information...	01100110...
Differential-based	A fascinating topic! Steganography is an ancient technique of hiding secret messages...	01100110...
random-sample	Steganography! A fascinating field that combines cryptography and computer science...	

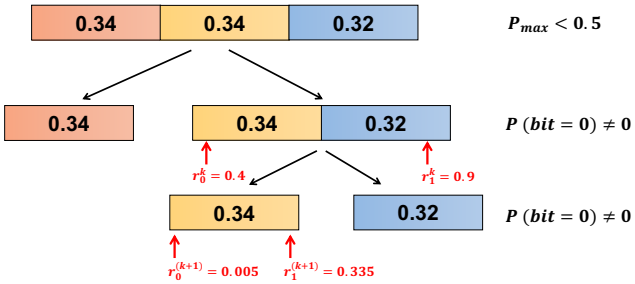

Figure 6: An illustration that Discop(Huffman) is not stable.

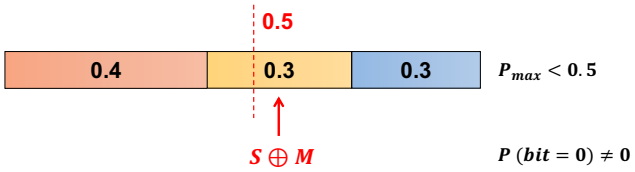

Figure 7: An illustration that Meteor is not stable.

E Experiments in security

We conducted experiments to verify the security of proposed schemes from steganalysis and KLD perspective[9] in Table 4.

Here the used steganalysis tool came from a classical and powerful work [41] for lingusitic steganalysis. We trained the classifier on a dataset using default settings and evaluated its performance on a test set, ultimately reporting the average results from ten repeated experiments. An accuracy range of 45% to 55% suggests that the steganalysis tool can hardly have crucial advantages against steganography. We also computed the frequency of stegotext under a uniform distribution and calculated the KLD between this and the expected distribution. The KLD approaching 0 suggests that the distribution of the stegotext is very close to that of the covertext, making it indistinguishable to an adversary.

[9]Meteor’s notable difference in KLD comes from the source code defaulting to discarding the least probable token.

Table 3: Resulting stegotext based on the history: “1+1=?”

	stegotext	hidden bits
Meteor	The answer is 2!	0
Meteor-reorder	The answer is 2!	0
Discop-base	The answer is... 2!	01
Discop-Huffman	The answer is 2!	01
Binary-based	The answer is 2!	0
Stability-based	The answer is 2!	01
Differential-based	2	0
random-sample	The answer is 2!	

Table 4: Experiments in security. Scenario A and B is tested by steganalysis tool (average accuracy and corresponding variance), while the Scenario C is tested by the KLD.

	Scenario A	Scenario B	Scenario C
Meteor	$54.94_{13.24}\%$	$54.12_{6.10}\%$	0.0337614
Meteor-reorder	$55.86_{14.09}\%$	$53.52_{5.87}\%$	0.0337614
Discop-base	$50.00_{2.75}\%$	$51.80_{5.64}\%$	0.0000356
Discop-Huffman	$49.95_{2.76}\%$	$51.86_{4.20}\%$	0.0000357
Binary	$49.83_{2.53}\%$	$51.41_{3.79}\%$	0.0000497
Stability	$49.73_{2.58}\%$	$52.12_{4.67}\%$	0.0000497
Differential	$49.66_{2.67}\%$	$53.00_{5.51}\%$	0.0000497
random-sample			0.0000435