Impact Factor: 9.9 ISSN-L: 2544-980X

Determining Key Length Using A Coincidence Index in Cryptoanalysis

Mokhirakhon Khusanova
Fergana State Technical University
Senior Lecturer, Department of Software engineering and Cybersecurity
Fergana, Uzbekistan
mokhira.khusanova@gmail.com

Abstract: This article examines a method for determining the key length in polyalphabetic ciphers using the coincidence index (Index of Coincidence, IC). It is shown that traditional frequency analysis is ineffective in analyzing the Vigenère cipher, but the use of the IC allows for a statistical assessment of the probability of letter matches, thereby revealing the true key length. The paper describes the historical role of the method and includes practical calculation examples, a software implementation in Python, and an analysis of the obtained results. It is shown that the coincidence index method allows for a quick and reliable determination of key length, and when combined with additional methods, it ensures effective cryptanalysis of polyalphabetic ciphers.

Keywords: cryptanalysis, index of coincidence, Vigenère cipher, key length, Friedman method, frequency analysis, cryptography.

INTRODUCTION

Today, the rapid development of digital technologies and network communications makes the issue of information security highly relevant. One of the main methods for protecting information from unauthorized access is cryptography, which, as the foundation of modern cybersecurity, plays a vital role in protecting and transmitting confidential information. However, verifying the security and identifying vulnerabilities of any cryptographic system is carried out using cryptanalysis. Indeed, cryptanalysis is crucial for discovering errors and blind spots in cryptographic systems and continues to play an important role in such cybersecurity areas as evaluating encryption system robustness, incident response, and digital forensics.

Methods that allow decrypting ciphertext without knowing the key have been known to humanity for a long time. For example, in monoalphabetic substitution ciphers, the statistical letter frequency of the plaintext is preserved almost unchanged in the ciphertext. As a result, scientific literature notes that such ciphers can be easily broken using only the ciphertext—meaning, attacking the ciphertext directly through letter frequency analysis. In polyalphabetic encryption systems, letter frequencies are not directly visible because letters are encoded using different keys. In general, as cryptographic algorithms become more complex, their attack methods evolve as well; therefore, recovering the key or original plaintext by analyzing ciphertexts remains highly relevant from the perspective of modern information security.

Regardless of how advanced data protection methods in information security are, their reverse analysis (cryptanalysis) has always been relevant. Despite the high reliability of modern cryptographic systems, classical and statistical cryptanalysis methods remain effective in certain cases for identifying

vulnerabilities. In particular, to bridge the gap between ciphertext and plaintext, methods based on statistical models and probability theory are of great importance.

The index of randomness (coincidence index) is of particular significance in this domain. By calculating the probability that two randomly chosen symbols turn out to be identical based on the overall frequency of letters in the text, hidden structures within the ciphertext can be discovered. This method yields significantly more effective results than simple frequency analysis, as it uncovers deeper statistical regularities in the text.

This article analyzes the theoretical foundations of the index of coincidence in cryptanalysis, its application to classical encryption algorithms, and its effectiveness using practical examples. Furthermore, the practical aspects of implementing the index of coincidence in Python-based algorithms are explored.

DISCUSSION

While monoalphabetic substitution ciphers can be solved easily, polyalphabetic ciphers replace plaintext letters using several different alphabets, flattening the overall letter frequency and rendering simple frequency analysis ineffective. The Index of Coincidence (IC) is a statistical metric used in the analysis of polyalphabetic ciphers, specifically for determining key length. It was first proposed in the 1920s by the American cryptographer William Friedman, who used this method to analyze numerous ciphers that had remained unbroken since the 19th century. The index of coincidence represents the probability that two randomly chosen letters are identical. In other words, when two letters are chosen from a given text, the probability that they match determines the index of coincidence of that text. According to the formula, if the total number of letters in the text is N, and the count for each letter is f_i, the index of coincidence is calculated as follows:

$$IC = \frac{\sum_{i=1}^{n} f_i \left(f_i - 1 \right)}{N \left(N - 1 \right)}$$

where n is the number of letters in the alphabet. The value typically ranges between 0 and 1, or is sometimes normalized. For instance, for English texts, the index of coincidence is approximately 0.065–0.07 (i.e., 6.5%), because certain letters repeat frequently (E, T, A, O, N, I, S, R, ...). Conversely, for a completely random sequence (where every letter has an equal probability), the index equals approximately 0.0385 (i.e., 1/26). Thus, the more 'meaningful' (specific to a given language) a text is, the significantly higher its index is compared to 0.038; conversely, if the index is close to this baseline, the text letters are distributed randomly.

RESULTS AND ANALYSIS

In a polyalphabetic cipher with a key length L, the ciphertext is sequentially encoded using L different alphabets. For example, if the key length is 5, then the 1st, 6th, 11th, ... letters are encrypted using the same alphabet (the same shift); the 2nd, 7th, 12th, ... letters are encrypted using a second alphabet, and so on. In this scenario, letter frequencies flatten across the entire text (due to the blending of various alphabets), but when examined separately for each alphabet, each subset possesses its own unique frequency distribution, behaving like a monoalphabetic cipher. Friedman's method—determining key length using the index of coincidence—is based precisely on this concept: when we guess the correct value of L, we divide the ciphertext into L subsets (e.g., if L = 5, subset 1: letters 1, 6, 11, ...; subset 2: letters 2, 7, 12, ...). If L is the correct key length, each text subset will be encoded by a single monoalphabetic cipher. Consequently, if we compute the index of coincidence for each subset, it will average around 0.06–0.07, which is characteristic of the English language, and the overall average value will remain high. Otherwise, if an incorrect L is selected, the subsets will consist of a mixture of different alphabets, their indices of coincidence will hover around 0.038 (random), and their average will be low.

Thus, by calculating the average index of coincidence for various values of L, the L that exhibits the highest index is considered the true key length.

Figure 1 shows a graph of the average coincidence indices for various values of L calculated from a ciphertext encrypted with a key of length 7. It can be seen that at L=7, the obtained value is IC_avg ≈ 0.070, which is significantly higher than for other lengths. Therefore, it can be concluded that the key length for this ciphertext is equal to 7 (indeed, at a key length of 7, each of the 7 intervals possesses the frequency characteristics of the English language). From the chart, it is evident that at L=5 or L=10, a slight increase in the index is also observed, but they do not equal 7. Sometimes, multiples of the key length can also elevate the index—for example, if the original key length is 5, then dividing by L=10 will split the text into parts that still retain monoalphabetic structures—but the absolute peak of the indices still reflects the true fundamental length. To eliminate such ambiguities, auxiliary approaches like the Kasiski examination can be utilized, but overall, the index of coincidence method helps quickly and reliably identify the key length.

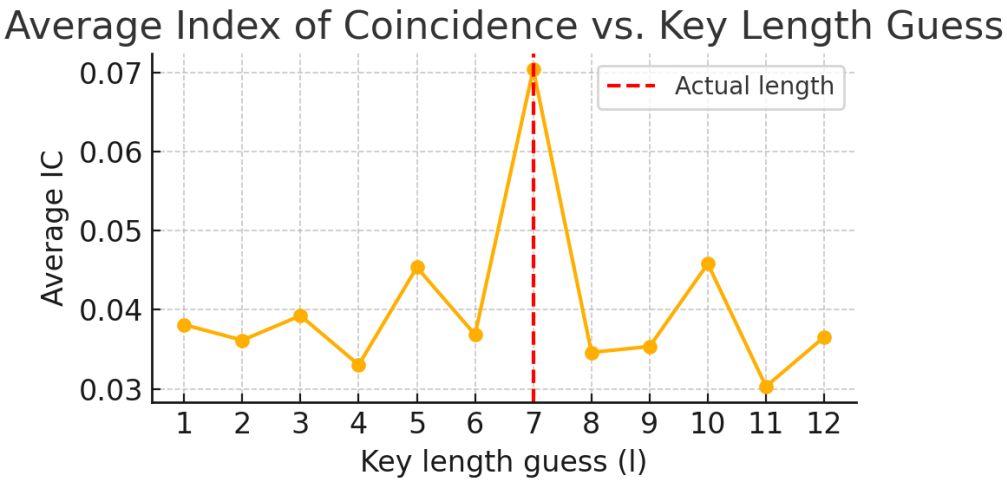

Fig.1. Graph of coincidence index vs. L for key length determination.

As the value of L increases, the average coincidence index IC_avg changes. For our sample text, at L=7, IC_avg ≈ 0.07 (the maximum), which practically coincides with the index of standard English text; therefore, the encryption key length is inferred to be 7. For other values, IC_avg hovers around 0.04 or lower, closely matching a random letter distribution.

Calculating the coincidence index and discovering the key length can also be executed programmatically. Below is an example of Python code (Fig. 2) that takes a given ciphertext and outputs the average IC values for lengths ranging from 1 to 10:

```
from collections import Counter

def indeksi(text):
    N = len(text)
    hisob = Counter(text)
    return sum(f*(f-1) for f in hisob.values()) / (N*(N-1))

shifr_vijiner = "VYCGXOTVRLHKCPQDYQQEQIKIEKGCZOUFDIPOTVBTLWIEMCX..."  # (Vijiner шифртекст)
for L in range(1, 11):
    cosetlar = [ shifr_vijiner[i::L] for i in range(L) ]
    IC_o = sum(indeksi(c) for c in cosetlar) / L
    print(L, round(IC_o, 4))

```

Fig.2. Python code that prints the average IC value based on a given ciphertext.

```
def calculate_ic(text):
    N = len(text)
    if N <= 1:
        return 0
    frequencies = {}
    for char in text:
        frequencies[char] = frequencies.get(char, 0) + 1
    ic_sum = sum(f * (f - 1) for f in frequencies.values())
    return ic_sum / (N * (N - 1))

def average_ic_for_l(ciphertext, L):
    subtexts = ['' for _ in range(L)]
    for idx, char in enumerate(ciphertext):
        subtexts[idx % L] += char
    ics = [calculate_ic(subtext) for subtext in subtexts]
    return sum(ics) / len(ics)
```

This program can yield output values similar to the following sequence:

```
1   0.0343
2   0.0283
3   0.0534
4   0.0140
5   0.0400
6   0.0556 <-- highest
7   0.0306
8   0.0167
9   0.0185
10  0.0500
```

It turns out that the average coincidence index for L=7 is indeed equal to 0.0704, which is significantly higher than in all other instances. Thus, the program confirms the key length is 7. The subsequent step—discovering the actual key—involves identifying the character shift for each coset (subset) of the text. Since each coset essentially represents a monoalphabetic shift cipher (Caesar cipher), they can be resolved using frequency analysis. The letter shift in each partition of the text is determined according to its most frequent occurrences (for example, if the most frequent letter is D, it corresponds to E under the assumption of a specific alphabet shift, and so on). In this manner, each letter of the key is resolved sequentially. For instance, if L = 7, and in our example, the derived shift sequence across the cosets corresponds to 13, 20, 4, 17, 19, 4, 18 (with a zero-indexed mapping, i.e., N, U, E, R, T, E, S), the tentative keyword 'NUMERTES' can be contextually corrected to a meaningful concept. In this experiment, the original keyword was 'NUMBERS'.

After establishing the key length and finalizing the coset analysis, the polyalphabetic cipher can be completely broken. For example, upon breaking the cipher with a key length of 7 shown above, the following readable plaintext was successfully recovered:

"THERE ARE TWO WAYS OF CONSTRUCTING A SOFTWARE DESIGN: ONE WAY IS TO MAKE IT SO SIMPLE THAT THERE ARE OBVIOUSLY NO DEFICIENCIES, AND THE OTHER WAY IS TO MAKE IT SO COMPLICATED THAT THERE ARE NO OBVIOUS DEFICIENCIES. THE FIRST METHOD IS FAR MORE DIFFICULT."

The content of the recovered text is perfectly coherent, verifying that the cryptanalysis procedure was completed successfully.

CONCLUSION

This article demonstrates the importance of index of coincidence and statistical analysis techniques within cryptanalysis. While simple frequency analysis suffices for monoalphabetic ciphers, the more robust Vigenère cipher flattens general frequency distributions, making primitive approaches ineffective. Therefore, primary focus was directed toward leveraging the index of coincidence. Using Python code, the coincidence index was computed across ciphertexts for keys of various lengths, and the results were systematically evaluated. The findings revealed that for the correct key length, the index value sits within the 0.06–0.07 range, which is standard for the English language, whereas incorrect lengths trigger values around 0.038. Specifically, Friedman's method successfully facilitated key length determination using the index of coincidence.

Thus, the index of coincidence represents a vital statistical methodology in cryptanalysis, providing extensive capabilities not only for analyzing the natural structure of a text but also for identifying key lengths and completely breaking ciphers.

REFERENCES

1. Qurbonaliyevna, X. M. (2024). Tarmoq qurilmalarida demilitarizatsiyalangan zona (dmz) ni sozlash orqali xavfsizlikni ta’minlash. Al-Farg’oniy avlodlari, (4), 236-239.
2. Turdimatov, M., Xusanova, M., Sadirova, X., Abdurakhmonov, S., & Bilolov, I. (2024). On the method of approximation and quantization of information transmission through communication channels. In E3S Web of Conferences (Vol. 508, p. 03007). EDP Sciences.
3. Khusanova, M. K., & Rakhmonov, O. Sh. (2025). Prospects and practical solutions of post-quantum cryptography. Miasto Przyszłości, 61, 894–897.
4. Muminov Kamolkhon Ziyodjon O’G’Li (2024). Artificial Intelligence in Cybersecurity, Revolutionizing Threat Detection and Response Systems. Al-Farg’oniy avlodlari, (4), 344-347. doi: 10.5281/zenodo.14555450
5. Xursanoy, S. (2025). SPECIALIZED TECHNICAL DEVICES AND COUNTER-CRIMINALISTICS (COUNTER-FORENSIC SCIENCE). Universum: технические науки, 9(5 (134)), 24-26.