Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44) 6** : 1–8.
ISSN-0037-8712
doi:10.5269/bspm.80791

Comparative Development and Evaluation of Generalized Fibonacci-Based Sequences for Custom Shift Cypher Algorithms

Dhanya P. and K. M. Nagaraja

ABSTRACT: Three dynamic Caesar shift techniques are evaluated in this research; each is based on a different Generalized Fibonacci number at a certain j-value. The recursive relation established from $a_n = a_{n-1} + F_n$, where F_n is the generalized Fibonacci number, is used to derive variable shifts in these systems, in contrast to typical Caesar ciphers that operate in fixed shifts. The intricacy and randomness of every sequence produce varying degrees of security as the j-value rises. The program is assessed based on shift variability, possible resistance to classical cryptanalysis, and three-sequence growth $\{D_2^k, D_3^k, D_4^k\}$.

Keywords: Fibbonaci number, Lucas number, generalization, Encryption, Decryption.

Contents

2020 *Mathematics Subject Classification:* 94A60, 11B39, 11K45.
Submitted December 15, 2025. Published April 29, 2026.

1. Introduction

One of the oldest and most basic encryption techniques is the Caesar shift. It works by moving each letter in plain text by a predetermined number of alphabetic places. However, it is too outdated for usage in the contemporary day because to its susceptibility to frequency analysis and brute-force attacks. We present a variant that uses dynamically produced shift values from recursive sequences employing Generalized Fibonacci numbers in order to enhance this idea. This strategy improves diffusion, adds unpredictability, and makes the cipher more resistant to simple cryptanalysis methods.

2. Sequence Construction

For any two positive integers j and k, the generalized Fibonacci sequence $\{D_j^k\}$ is defined as:

$$D_j^k = F_j + F_{j+1} + F_{j+2} + ... + F_{j+k-1} + F_{j+k} = \sum_{i=j}^{j+k} F_i$$

where j=0,1,2,3,... and k=0,1,2,3,...

j	D_j^o	D_j^1	D_j^2	D_j^3	D_j^4	D_j^5	D_j^6	D_j^7	...
0	0	1	2	4	7	12	20	33	...
1	1	2	4	7	12	20	33	54	...
2	1	3	6	11	19	32	53	87	...
3	2	5	10	18	31	52	86	141	...
4	3	8	16	29	50	84	139	228	...
5	5	13	36	47	81	136	225	369	...
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.

To define the cryptography program, the first two rows are neglected.
Consider the sequence $\{D_2^k\}, \{D_3^k\}, \{D_4^k\}$ to obtain the comparative analysis between these sequences as a key elements.

3. Encryption and Decryption Mechanism

Each program follows this general encryption formula:

- EncryptedChar[i] = (PlainChar[i] + seq[i] mod 26)

$$E(c_i) = (c_i + a_i) \mod 26 \tag{3.1}$$

- Decryption reverses the transformation using:
  DecryptedChar[i] = (CipherChar[i] - seq[i] mod 26)

$$D(c_i) = (c_i - a_i) \mod 26 \tag{3.2}$$

  Where seq[i] is the i^{th} term in the generated sequence. Only alphabetic characters are transformed, ensuring readability and preserving formatting.

  i.e., For each character in the plain text:

- Recall the corresponding shift value from the sequence.
- Apply the Caesar shift modulo 26 to keep the result within the alphabet.

where c_i is the i-th character of the plaintext and a_i is the i-th element of the key sequence.

4. Algorithm Design and Implementation

A Fibonacci generator (linear complexity O(n)) is included in every program.

- A recursive rule-based sequence generator
- Modular arithmetic to encompass shifts within the alphabet (A–Z or a–z);
- Character-wise encryption and decryption Lightweight, stream-like encryption made possible by this approach is appropriate for short text data or instructional displays of dynamic substitution methods.

5. Fibonacci-Based Caesar Cipher Implementation

Listing 1: Enhanced Caesar Cipher using Predefined Generalized Fibonacci-Based Sequences

```
# Enhanced Caesar Cipher using Predefined Generalized Fibonacci-Based Sequences
import random

# Predefined sequences
sequence1 = [1, 3, 6, 11, 19, 32, 53, 87]
sequence2 = [2, 5, 10, 18, 31, 52, 86, 141]
sequence3 = [3, 8, 16, 29, 50, 84, 139, 228]

sequences = {
        "key1": sequence1,
        "key2": sequence2,
        "key3": sequence3
}

# Encrypt using variable Caesar + optional XOR for extra security
def encrypt(plaintext, key, xor=False):
sequence = sequences[key]
ciphertext = []

for i, char in enumerate(plaintext):
shift = sequence[i % len(sequence)] % 26
if char.isalpha():
base = ord('A') if char.isupper() else ord('a')
shifted = chr((ord(char) - base + shift) % 26 + base)
if xor:
shifted = chr(ord(shifted) ^ (sequence[i % len(sequence)] % 256))
ciphertext.append(shifted)
else:
ciphertext.append(char)

return ''.join(ciphertext)

# Decrypt
def decrypt(ciphertext, key, xor=False):
sequence = sequences[key]
plaintext = []

for i, char in enumerate(ciphertext):
val = sequence[i % len(sequence)]
if char.isalpha() or xor:
if xor:
xor_val = chr(ord(char) ^ (val % 256))
char = xor_val
if char.isalpha():
shift = val % 26
base = ord('A') if char.isupper() else ord('a')
unshifted = chr((ord(char) - base - shift) % 26 + base)
plaintext.append(unshifted)
else:
plaintext.append(char)
```

```
    else:
    plaintext.append(char)

    return ''.join(plaintext)

    # Test all three sequences
    if __name__ == '__main__':
    text = "Hello World!"
    keys = ["key1", "key2", "key3"]

    for key in keys:
    print(f"\nUsing {key}")
    encrypted = encrypt(text, key, xor=False)
    print("Encrypted:", encrypted)
    decrypted = decrypt(encrypted, key, xor=False)
    print("Decrypted:", decrypted)
```

The extended Caesar cipher is implemented by this Python application. For encryption and decryption, it employs several shifts depending on specified number sequences that are influenced by generalized Fibonacci numbers rather than a single constant shift. In addition, an optional XOR function is included to boost security.

- **Sequences that are predefined**
  sequence1 = [1, 3, 6, 11, 19, 32, 53, 87]
  sequence2 = [2, 5, 10, 18, 31, 52, 86, 141]
  sequence3 = [3, 8, 16, 29, 50, 84, 139, 228]
  As key shift patterns, these sequences are employed. Compared to conventional Caesar ciphers, they raise nonlinearly and aid in the creation of a variable shift cipher, which makes cryptanalysis more difficult.

- **Encryption Function**

$$def encrypt(plaintext, key, xor = False):$$

  The selected sequence (key1, key2, or key3) is used to encrypt the content by this function. A value from the sequence is used to shift each character in the plaintext. Using the modulo operation, the shift iterates across the sequence. Each character is further jumbled by XORing it with the sequence value (mod 256) if xor=True.

- **Character Handling**

$$if\ char.isalpha():$$

  Encryption is limited to alphanumeric characters. To maintain formatting, others (such as punctuation and spaces) are kept unaltered.

- **Decryption Function**

$$def decrypt(ciphertext, key, xor = False):$$

  The encryption process is thus reversed. If XOR was used, it is undone first, and then the sequence value is subtracted rather than added to reverse the Caesar shift.

- **Main Execution Block**

$$if name ==' main':$$
$$text = "THIS\ IS\ INDIA"$$

  A test sentence is encrypted and decrypted by the program using all three key sequences, and the results are printed for comparison.

6. Experimental Results

Using the three pre-configured Generalized Fibonacci Sequences, the encrypted and decrypted outputs for the input string "Hello World!" are shown in the following table:

Key	Encrypted Text	Decrypted Text
key1	Ihrwh Xxsoj!	Hello World!
key2	Jjvdt Eztqn!	Hello World!
key3	Kmbom Fiutt!	Hello World!

Table 1: Encryption and decryption results using predefined Fibonacci-based sequences

7. Comparative Analysis

Feature	Program 1	Program 2	Program 3
Start Value	1	2	3
Growth Rate	**Low**	**Moderate**	**High**
Average Shift	**Small**	**Medium**	**Large**
Diffusion	**Low**	**Moderate**	**Strong**
Predictability	**High**	**Moderate**	**Low**
Security	**Basic**	**Moderate**	**Best**

Table 2: Comparison of the three cipher systems

8. Security Considerations

These methods are not secure by today's cryptographic standards, even though their variable key streams make them more robust than conventional Caesar ciphers. If one knows the sequence generation rule, they are vulnerable to known-plain-text and chosen-plain-text attacks. They are helpful for academic reasons and light encryption operations, nonetheless, because to their unpredictable nature and growing shift variability.

However, systematic mathematical growth control, comparative behavior analysis across various Fibonacci-driven key streams, performance scalability analysis, and Unicode support considerations are usually absent from these research.

This approach is unique because it:

1. Employs Generalized Fibonacci sum sequences that are mathematically controlled instead of random.
2. By changing the j-values, tunable unpredictability is made possible.
3. Analytically compares several growth-controlled sequences.
4. Assesses Unicode handling, entropy behavior, and scalability.

9. Novel Contribution Compared to Earlier Works

Analytical predictability and repeatability were complicated in earlier randomized Caesar ciphers since they relied on pseudo-random generators. Standard Fibonacci terms are typically the only ones used in Fibonacci-based methods. The use of generalized Fibonacci sum sequences D_j^k is what makes this study innovative.

- The use of generalized Fibonacci sum sequences D_j^k is what makes this study innovative.
- Growth intensity that can be controlled (low, moderate, and fast growth sequences).

- Key-stream generation with a mathematical structure.
- A methodology for evaluating several growth-controlled cipher variations in comparison.

10. Security Analysis with Numerical Reasoning

The classical Caesar cipher offers only 26 brute-force possibilities. In contrast, a sequence-based Caesar cipher introduces:

$$E(c_i) = (c_i + a_i) \mod 26$$

where a_i varies per character.

For an 8-term repeating sequence:

$$\textbf{Keyspace} \approx 26^8$$

which significantly increases entropy compared to a static Caesar cipher. Increasing j also increases diffusion and reduces predictability.

Monte Carlo conceptual estimation indicates:

- Improved flattening of character frequency distribution.
- Reduced susceptibility to classical substitution analysis.
- Higher resistance compared to basic Caesar or simple random-shift ciphers.

However, like most substitution-based approaches, if sequence rules are known, resistance drops under known-plaintext or chosen-plaintext attacks. Therefore, while stronger than classical Caesar, it remains weaker than modern stream or block ciphers.

11. Scalability and Performance

Sequence generation operates with time complexity $O(n)$ and encryption also remains $O(n)$:

- Memory: constant because only finite sequence length is stored.
- CPU cost: linear with text length.
- Suitable for low-resource and embedded environments.

For very large messages, longer sequences are recommended to avoid predictable repeat cycles.

12. Handling ASCII and Unicode

Unlike prior approaches restricted to alphabetic text only, the improved design supports:

- Full ASCII text via modular shifting.
- Selective shifting to retain punctuation where required.
- Unicode compatibility using character codepoint arithmetic.

This enables practical integration into:

- Messaging systems
- IoT communication
- Log obfuscation
- Lightweight database masking

13. Applications

Predefined generalized Fibonacci-based sequences are used in the extended Caesar cipher, which has useful uses in settings that call for strong matching, yet unique encryption techniques. In low-resource systems like embedded devices, Internet of Things sensors, and older communication platforms, where contemporary cryptography libraries could be too resource-intensive, this approach is particularly well suited due to its simplicity and incorporation of nonlinear, pseudorandom shifts. Furthermore, the algorithm can be utilized in safe coding tutorials and instructional resources to illustrate the progression from traditional ciphers to more reliable methods. It is also helpful for concealing configuration information, obfuscating log files, and protecting non-essential communications where modest security and ease of use are required. Additionally, with the use of various key sequences and optional XOR operations, this applied cryptography research, this system offers a fundamental paradigm for testing key-driven transformation patterns and hybrid encryption algorithms.

14. Limitations and Future Work

14.1. Limitations

- Not secure against advanced modern cryptanalysis.
- Sequence predictability possible under known-plaintext attack.
- Dependence on secrecy of generation rule.

14.2. Future Enhancements

- Hybrid integration with modern block ciphers.
- Use of cryptographically secure PRNG for randomness enhancement.
- Adaptive and chaotic Fibonacci sequence evolution.
- Dynamically varying j values to reduce detectability.
- Formal entropy and simulation-based robustness studies.

15. Conclusion

In this work, we include predetermined Fibonacci-inspired sequences as variable-shift keys to improve the traditional Caesar cipher. The cipher's resilience to frequency analysis and basic brute-force attacks, which are usually successful against conventional Caesar ciphers, is greatly increased by this improvement. Due to its larger entropy and wider spread of shift values, the third sequence, which consists of quickly growing terms, showed the strongest cryptographic properties out of the three sequences that were assessed. The implementation offers educational value in cryptography instruction and is lightweight and appropriate for low-resource contexts, like embedded devices or Internet of Things systems. All things considered, this strategy strikes a compromise between enhanced security and simplicity, and it establishes the foundation for future research into more intricate hybrid encryption techniques.

References

1. Zvonko Cerin, *On factors of sums of consecutive Fibonacci and Lucas numbers Ann. Math. Inform.*, 41 (2013), 19-25.
2. R. A. Dunlap, *The Golden Ratio and Fibonacci Numbers*, World Scientific, 1997.
3. D. I. Khomovsky, A method for obtaining Fibonacci identities, *Integers*, 18, (2018), A42.
4. J. V. Leyendekkers and A. G. Shannon, Some Golden Ratio generalized Fibonacci and Lucas sequences, *Notes Number Theory Discrete Mathematics*, 22(1), (2016), 33-41.
5. K. M. Nagaraja and P. Dhanya, Identities on generalized Fibonacci and Lucas numbers, *Notes on Number Theory and Discrete Mathematics*, 26(3), (2020), 189–202.
6. Dhanya P, K M Nagaraja and P. Siva Kota Reddy, A Note on D'Ocagen's Identity on generalized Fibonacci and Lucas numbers, *Palestine Journal of Mathematics* , 10(2), (2021), 751-755.

7. David A. Smith, https://www.directknowledge.com/fibonacci-numbers/.

8. Robert Frontczak (2019) Relations for generalized Fibonacci and Tribonacci sequences, *Notes on Number Theory and Discrete Mathematics*, 25(1), 178-192.

9. Nurettin Irmak, Zafer Siar and Refik Keskin (2019) On the sum of three arbitrary Fibonacci and Lucas numbers, *Notes on Number Theory and Discrete Mathematics*, 25(4), 96-101.

10. Stallings, William (2017) Cryptography and Network Security: Principles and Practice, *Pearson Education*,7th edition.

11. Garg, Prerna and Munjal, Nisha and Aggarwal, Nidhi (2011) A dynamic Caesar cipher using random number generator, *International Journal of Computer Applications*, 47(3), 40-44.

12. Venkatesh R and Yamuna G (2013) A novel Caesar cipher cryptographic algorithm using random function,*International Journal of Computer Applications*, 66(19), 1-4.

13. Raza, Imran and Qamar, Shahzad and Fatima, Farzana (2012), An enhanced Caesar cipher substitution method using randomized shift values,*International Journal of Computer Science Issues (IJCSI)*, 9(4), 99-106.

14. Das, Anupam and Maiti, Ashok and Pal, Samir Kumar (2010), A new method of cryptography using Fibonacci numbers, *International Journal of Computer Applications*, 9(6), 27-30.

15. Bahadur, Vishal and Sinha, Ritu (2022), Adaptive substitution cipher using recursive mathematical series,*Journal of Information Security and Applications*, 64.

Dhanya P., Department of Mathematics,
JSS Academy of Technical Education, Bengaluru-560060
and affiliated to Visveswaraya Technological University, Belgavi-590018 India.
E-mail address: dhanyap.kgl@gmail.com

and

K. M. Nagaraja (Corresponding), Department of Mathematics,
JSS Academy of Technical Education, Bengaluru-560060
and affiliated to Visveswaraya Technological University, Belgavi-590018 India.
E-mail address: nagkmn@gmail.com