RPA-Code for Secure Binary Sequence Generation from Graph-Based Scanning

Ahcène Bounceur, Mohammad Hammoudeh, Jameleddine Hassine, Madani Bezoui, Adel Fadhl Ahmed, Muhamad Felemban

RPA-Code for Secure Binary Sequence Generation from Graph-Based Scanning

Ahcene Bounceur[1,2,3], Mohammad Hammoudeh[1,2], Jameleddine Hassine[1,2], Madani Bezoui[4], Adel Fadhl Ahmed[1], Muhamad Felemban[1,2]
ahcene.bounceur@kfupm.edu.sa
[1]Information and Computer Science Department, KFUPM, Saudi Arabia
[2]Interdisciplinary Research Center for Intelligent Secure Systems, KFUPM, Saudi Arabia
[3]Interdisciplinary Research Center for Intelligent Manufacturing and Robotics, KFUPM, Saudi Arabia
[4]CESI - LINEACT, Nice, France

ABSTRACT

This article introduces a novel method for generating random binary sequences from Random Polar Angles (RPA). These sequences can be derived from an image, akin to QR-Codes, making them suitable for cryptographic applications and information coding systems. The proposed method allows the generation of multiple codes using the same image. It is based on the LPCN (Lowest Polar Angle Connectivity Node) algorithm, which, in each iteration, selects a node from a graph that forms the minimal polar angle with its neighbor found in the previous iteration. The decision on the type of bit to generate (0 or 1) is based on the nature of the angles found (acute or obtuse). The primary objective of this research is to facilitate the transformation of a graph's image representation into a concealed binary code, thereby enhancing data security. We have assessed the randomness of the generated codes through entropy analysis, revealing greater unpredictability.

KEYWORDS

Random generation, graph, polar-angle, LPCN algorithm, key generation

ACM Reference Format:
Ahcene Bounceur[1,2,3], Mohammad Hammoudeh[1,2], Jameleddine Hassine[1,2], Madani Bezoui[4], Adel Fadhl Ahmed[1], Muhamad Felemban[1,2]. 2018. RPA-Code for Secure Binary Sequence Generation from Graph-Based Scanning. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Generating random numbers from pictures is an emerging field with applications in various domains, including cryptography, security, and data encryption. In today's world, QR codes have become the most commonly employed method for copying information without the need for manual data entry or keyboard input. For publicly accessible, non-sensitive information, QR codes can be utilized without any concerns. However, when these codes are used to conceal or safeguard sensitive data, as far as we are aware, there is currently no established standard code designed to address this particular requirement. In this paper, we introduce a novel approach known as Random-Polar-Angle (RPA) coding for generating random codes that are inherently resistant to easy decipherment.

Generating random numbers from pictures is an evolving field, with a wide range of potential applications. Researchers and practitioners continue to explore new methods and techniques to harness the inherent randomness present in visual data.

Cryptography relies heavily on the generation of secure and unpredictable cryptographic keys. Traditional methods often use pseudorandom number generators, but the security of such sequences can be compromised. Over the years, researchers and cryptographic practitioners have delved into a multitude of techniques and algorithms to produce keys and binary streams that are both highly secure and unpredictable. This pursuit of cryptographic robustness has led to the development of innovative methods that harness mathematical principles, complex algorithms, and even physical phenomena. One such approach revolves around the use of graph-based algorithms, where the structure of a graph is utilized to derive secure binary sequences. In this context, the LPCN (Lowest Polar Angle Connectivity Node) algorithm, as introduced in [14][19], stands as a prominent example. By selecting nodes within a graph based on minimal polar angles, the LPCN algorithm introduces an element of randomness and complexity that enhances the security of generated binary sequences. This technique has gained attention for its potential to transform graphical representations into concealed binary codes, thereby bolstering data protection, by transforming the type of the found angle in each iteration to a binary value such as:

- acute angle = 1
- obtuse angle = 0

The rest of the paper is organized as follows. Section 2 presents and discusses the related work. Section 3 describes the LPCN algorithm. The proposed methodology is presented in Section 4. Section 5 presents the implementation and results obtained by applying the proposed method, followed by an assessment of the randomness of the generated sequences. Finally, Section 6 concludes the paper and offers perspectives for further exploration.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

2 LITERATURE REVIEW

There has been a significant amount of work done in the domain of generating random numbers from images. Image steganography [5] is the practice of embedding hidden messages or data within digital images without altering their visible appearance. This technique is used for covert communication and information security. Chandramouli and Memon [3] investigated the "*steganographic capacity*" of image based steganography techniques. Steganographic capacity refers to the maximum amount of data that can be concealed or embedded within an image without causing noticeable or statistically significant changes to that image. It is typically measured in terms of bits. In other words, it answers the question of how much information can be hidden in an image without making the presence of the hidden data evident. The authors [3] analyzed LSB (Least Significant Bits) steganography, which is a specific method for achieving steganographic capacity. In LSB steganography, the hidden data is inserted into the least significant bits of the pixels in an image, while the most significant bits remain unchanged. Since the least significant bits have the least impact on the visual appearance of the image, this method aims to maximize the steganographic capacity while minimizing the noticeable alterations to the cover medium. In order to increase information security, Wang et al. [21] introduced a hybrid steganography method based on least significant bit (LSB) replacement and Hamming code (HLAH). As the sharp areas of the image can accommodate greater alterations compared to the smooth areas, the HLAH method embeds a higher volume of secret messages in edge regions and a minimal amount in smooth regions. Experimental results [21] demonstrate that the HLAH method offers both superior embedding capacity compared to existing techniques and consistently maintains higher image quality. Cheddad et al. [4] presented a state-of-the-art review and analysis of the different existing steganography techniques. More recently, the utilization of Generative Adversarial Networks (GANs) [6] in image steganography research has shown significant promise and potential. The GAN [6] may be trained adversarially to ensure that the stego-images are visually identical to the cover images. This adversarial training makes it challenging for external parties to detect the presence of hidden data. The generator in a GAN [6] can be used to embed the secret information into an image. Instead of generating random images, the generator can take the cover image and encode the hidden data in a way that the output image appears visually similar to the original one. This process ensures that the embedded information is perceptually indistinguishable from the original image. The GAN discriminator is used to ensure that the generated stego-image appears similar to real images and not distinguishable from authentic images. It helps in maintaining the quality of the stego-image. Liu et al. [17] conducted an extensive review of steganography techniques utilizing GANs, systematically categorizing them into three distinct data hiding strategies, encompassing cover modification, cover selection, and cover synthesis. For a comprehensive survey of image steganography, the reader is referred to [10].

In pixel-based methods, the randomness is extracted from the pixel values of an image. The noise in an image, such as that from a camera sensor or compression artifacts, can be a source of randomness. Techniques involve analyzing pixel values for entropy, employing hash functions, or applying statistical tests to determine the randomness level. Hu and Han [8] introduced an efficient and secure pixel-based scrambling method for safeguarding the distribution of digital medical images. In order to efficiently encrypt a substantial volume of digital medical images, their approach employs a straightforward pixel-level XOR operation for image scrambling, incorporating structural parameters of the encryption scheme into the cryptographic key. This cryptographic key is derived from a genuinely random number sequence generated from multi-scroll chaotic attractors.

Visual Cryptography [9] is a technique that encrypts an image into multiple shares, each containing partial information. Combining these shares reveals the original image. The randomness in these shares can be harnessed as a source of random numbers. Gurunathan and Rajagopalan [7] drew inspiration from steganography and introduced a technique for concealing a secret message within a cover image. This method divides the cover image into n blocks, each measuring *8x8* pixels, while the secret message is partitioned into n segments. This approach enhances both image quality and the capacity of the hidden message, bolstering its security. In their proposed method [7], they utilize Cuckoo Search (CS) to search for approximate and optimal solutions for transforming each block of the message. This is in contrast to seeking a single optimal substitution matrix for the entire cover image."

Random sequence generators typically fall into two categories: true random number generators (TRNGs) and pseudorandom number generators (PRNGs). PRNGs rely on seeds and deterministic algorithms to produce "pseudorandom" numbers, making them faster and suitable when a significant quantity of random-like numbers is needed. In contrast, TRNGs harness non-deterministic sources in conjunction with post-processing functions, such as image noise, to generate randomness. The resulting sequences from TRNGs are recognized for their heightened level of randomness. Zhou et al. [22] proposed an innovative True Random Number Generator (TRNG) that derives random bits from mouse movements, which are captured as a 2D image. This method offers a practical, versatile, and cost-effective solution for personal computer (PC) platforms. To mitigate regular patterns in mouse movements stemming from user habits, the authors proposed three TRNGs based on chaotic hash functions. Comprehensive experiments were carried out to assess the speed, diffusion, and randomness characteristics of these TRNGs. The results indicate that two of these TRNGs exhibit satisfactory performance and are suitable for implementation on standard PC platforms. Dragan and Mladen [15] introduced a method for generating pseudo-random numbers using a discrete-space chaotic map based on permutation compositions. The randomness of the generated pseudo-random sequences is verified through NIST 800-22 and TestU01 tests. Notably, this method [15] remains unaffected by dynamical degradation, ensuring stable pseudo-random number generation. Its advantages include an extensive key space, efficient memory usage, enhanced security, and extended cycle lengths, making it suitable for memory-constrained devices. Lakshman et al. [13] introduced an innovative and straightforward pseudo-random number generation method that utilizes one-dimensional discrete chaotic maps and a logo image. The method combines sequences from chaotic maps and a logo image to generate real numbers within the range of zero to one,

establishing a strong connection between the two. This approach yields numbers that exhibit high sensitivity to minor variations in the underlying keys. The effectiveness of the generator is assessed through a comprehensive set of statistical and security analyses. Rongzhong Li [16] introduced a true random number generator (TRNG) approach using the images taken by web or mobile phone cameras. The proposed TRNG technique uses all three RGB color channels to obtain the random numbers. The study also investigated the physical and statistical properties of the random noise in a digital images and made several approximations to efficiently collect the best random signals from the pixels in the images to map them to random sequences.

Amri et al. [2] introduced a novel quantum random number generation method leveraging the inherent randomness of photon emissions and the single photon counting capability of the Quanta Image Sensor (QIS). The QIS, consisting of over one billion pixels known as 'jots,' is optimized for photon counting with sub-micron pitch. This high-speed, low-power QIS technology has the potential to generate high-quality random numbers at a remarkable data output rate, surpassing conventional QRNG technologies.

Kawashima and Mito [12] proposed a method for generating random numbers from magnetic domain patterns within a magneto-optical (MO) material. By capturing one hundred thousand chaotic magnetic domain images using a transparent polarization microscope, they applied post-processing techniques, including binarization, frequency adjustment, exclusive-OR operations, and offset data reduction, to extract random numbers from the images. These generated random numbers successfully passed the statistical test (NIST SP-800-22).

Yas Abbas Alsultanny [1] introduced a 3-phase approach for generating a highly random bit sequence. In the initial phase, image data is read and partitioned into blocks. The second phase is dedicated to generating a random key, and the third phase conducts statistical tests to assess whether the key sequence exhibits the characteristics expected of a genuinely random sequence. These tests are probabilistic in nature, not deterministic. Users can choose to accept the generated key based on a threshold value, indicating the minimum number of successful tests required. If the key doesn't meet this criterion, the process can be repeated until a satisfactory key is obtained. The algorithm employs five statistical tests: Frequency, Serial, Poker, Runs, and Auto-correlation. According to the author [1], 79% of the generated sequences successfully pass all five statistical tests concurrently, with 94% passing at least four of them.

3 LPCN ALGORITHM

The LPCN (Least Polar-angle Connected Node) algorithm is used to find a polygon hull of a set of connected points. It starts from a node that belongs to the border. It is given as follows. Let V be a set of n nodes of a network and h be the number of points of the polygon hull of V. The algorithm finds the points P_0, P_1, ..., P_{h-1} of the convex hull. First of all, we find a point P_0 with the minimum x-coordinate. Given the point P_i, we wish to find the next consecutive point P_{i+1} on the hull, for each $1 \leq i \leq k-1$ which is connected to them. This point P_{i+1} is the one that has the least polar angle with respect to P_i. When we reach the highest point P_{h-1}, we have constructed the polygon hull of V. The algorithm stops once we reach again the point P_0. It is given as follows:

procedure LPCN1(V, E)
  $P_c \leftarrow$ point having the minimum x-coordinate
  $\mathbb{B} \leftarrow \{P_c\}$
  $P_f \leftarrow P_c$
  $P_p \leftarrow$ fictive point situated in the left of P_f
  repeat
    P_v *is such that* $pa(P_p, P_c, P_v) =$
      $\min\{pa(P_p, P_c, P_j) :$
      $P_j \in \aleph(P_c)$ & $P_j \notin B - \{P_f\}\}$
    $\mathbb{B} \leftarrow \mathbb{B} \cup \{P_v\}$
    $P_p \leftarrow P_c$
    $P_c \leftarrow P_v$
  until $P_v = P_f$
  return $\mathbb{B}$
end procedure

Figure 1: LPCN1 algorithm (version 1).

Figure 3 shows how the algorithm works. Let us consider the graph of Figure 3(a) where $V = \{A, B, C, D, E, F, G, H\}$. First, we choose the node that has the minimum x-coordinate P_c. In this example, this node is A ($P_c = A$) which is the first node of the border $\mathbb{B} = \{A\}$.

The end of the algorithm is determined when the next border node P_k is equal to the first border node P_f. In this example, the first node is also A ($P_f = A$). In the first iteration P_c is always equal to P_f.

Let consider a fictive node A' that has an x-coordinate smaller than that of A ($P_p = A'$). Note, that other fictive nodes can be considered. Figure 3(b) shows the different starting nodes and their fictive neighbors designed by the gray points and letters with apostrophes.

Next, we have to find the minimum polar angle formed by the edge (P_c, P_p) (i.e., (A, A')) and the edge formed by P_c and each edge of its neighbors $\aleph(P_c)$ (i.e., (A, B), (A, C), (A, D) and (A, G)). In this example, the obtained neighbor is $P_k = C$ (cf. Figure 3(c)). Then, the second edge of the searched border is (A, C). Therefore, $\mathbb{B} = \{A, C\}$.

In the next iteration we will do the same procedure by searching the minimum polar angle formed by the edge (C, A) and edges formed by C and its neighbors, i.e., (C, B) and (C, D) except (C, A), the last found edge. The obtained node is D (cf. Figure 3(d)). Therefore, the obtained border is $\mathbb{B} = \{A, C, D\}$. In the same way we determine the other nodes. We found E (cf. Figure 3(e)), H (cf. Figure 3(f)), G (cf. Figure 3(g)) and finally A (cf. Figure 3(h)). Since $A = P_f$, we stop the algorithm. The obtained border is $\mathbb{B} = \{A, C, D, E, H, A\}$ (cf. Figure 3(i)).

The first version of the algorithm LPCN can not work in some graphs when and edge in the border intersect another edge of the border. In case where the aim is to determine the border nodes, this version can not be used, however, in the framework of this paper, we can use it since the objectif is not to find the border nodes but a sequence of nodes that can be used as a code. The following algorithm 2 presents the second version of the LPCN algorithm

(LPCN2) where the intersection is eliminated. It will be used in this paper to compare and discuss the codes obtained from both versions.

For simplicity, by considereing only the three first cases, the proposed algorithm is defined as follows:

1: **procedure** LPCN2(V, E)
2: $P_c \leftarrow$ point having the minimum x-coordinate
3: $\mathbb{B} \leftarrow \{P_c\}$
4: $P_f \leftarrow P_c$
5: $P_p \leftarrow$ fictive point situated in the left of P_f
6: **repeat**
7: **if** $|\aleph(P_c)| > 1$ **then**
8: $\mathbb{A} = \{P_p, P_f\}$
9: **else**
10: $\mathbb{A} = \{P_f\}$
11: **end if**
12: P_v *is such that* $pa(P_p, P_c, P_v) =$
13: $\min\{pa(P_p, P_c, P_j) : P_j \in \aleph(P_c)\ \&\ P_j \notin \mathbb{A}\}$
14: **if** intersection detected **then**
15: $\mathbb{A} = \mathbb{A} \cup \{P_v\}$
16: Go to 12
17: **end if**
18: $\mathbb{B} \leftarrow \mathbb{B} \cup \{P_v\}$
19: $P_p \leftarrow P_c$
20: $P_c \leftarrow P_v$
21: **until** $P_v \neq P_f$
22: **return** $\mathbb{B}$
23: **end procedure**

Figure 2: LPCN2 algorithm (version 2).

The LPCN algorithm demonstrates minimal energy consumption [11][18], and its computational complexity is polynomial.

4 METHODOLOGY

In this section, we will present the proposed methodology. First, a graph will be generated randomly by considering three main parameters:

- The area a : which represents a rectangular space where the graph will be generated
- The number of nodes n of the graph
- The range r which represents the circular area around a given node such that all the nodes belonging to the circle will be connected to this node

Figure 4 shows an example of a linear graph with $n = 8$ nodes. When we apply the LPCN algorithm to this graph, starting from node 1, we traverse the following angles. Additionally, each angle's type and corresponding binary value are provided.

- $\widehat{(1,2,3)}$: obtuse = 1
- $\widehat{(2,3,4)}$: acute = 0
- $\widehat{(3,4,5)}$: acute = 0
- $\widehat{(4,5,6)}$: obtuse = 1
- $\widehat{(6,7,8)}$: obtuse = 1
- $\widehat{(7,8,9)}$: obtuse = 1

This will lead to the generation of the binary stream 100111. The proposed methodology is composed of two main steps:

- Image Transformation: One of the primary applications of this method is the transformation of an image of the graph into a hidden binary code. By scanning the graph in the image and applying the LPCN algorithm, a binary sequence is generated. This sequence can be embedded into the image in a way that is imperceptible to the human eye, ensuring data security.
- The Graph Representation: The first step involves representing the data as a graph, where nodes represent data points or features, and edges represent connections between them. This graph is used as the foundation for generating the binary sequence.
- LPCN Algorithm: The LPCN algorithm is employed to traverse the graph and select nodes. At each iteration, the algorithm identifies the node that forms the minimal polar angle with its neighbor from the previous iteration. This unique selection process introduces randomness and complexity into the sequence generation.
- Iterative Selection: The LPCN algorithm iteratively selects nodes, forming a sequence of nodes from the graph. These selected nodes are then transformed into binary values, ensuring that the generated binary sequence is both secure and unpredictable.

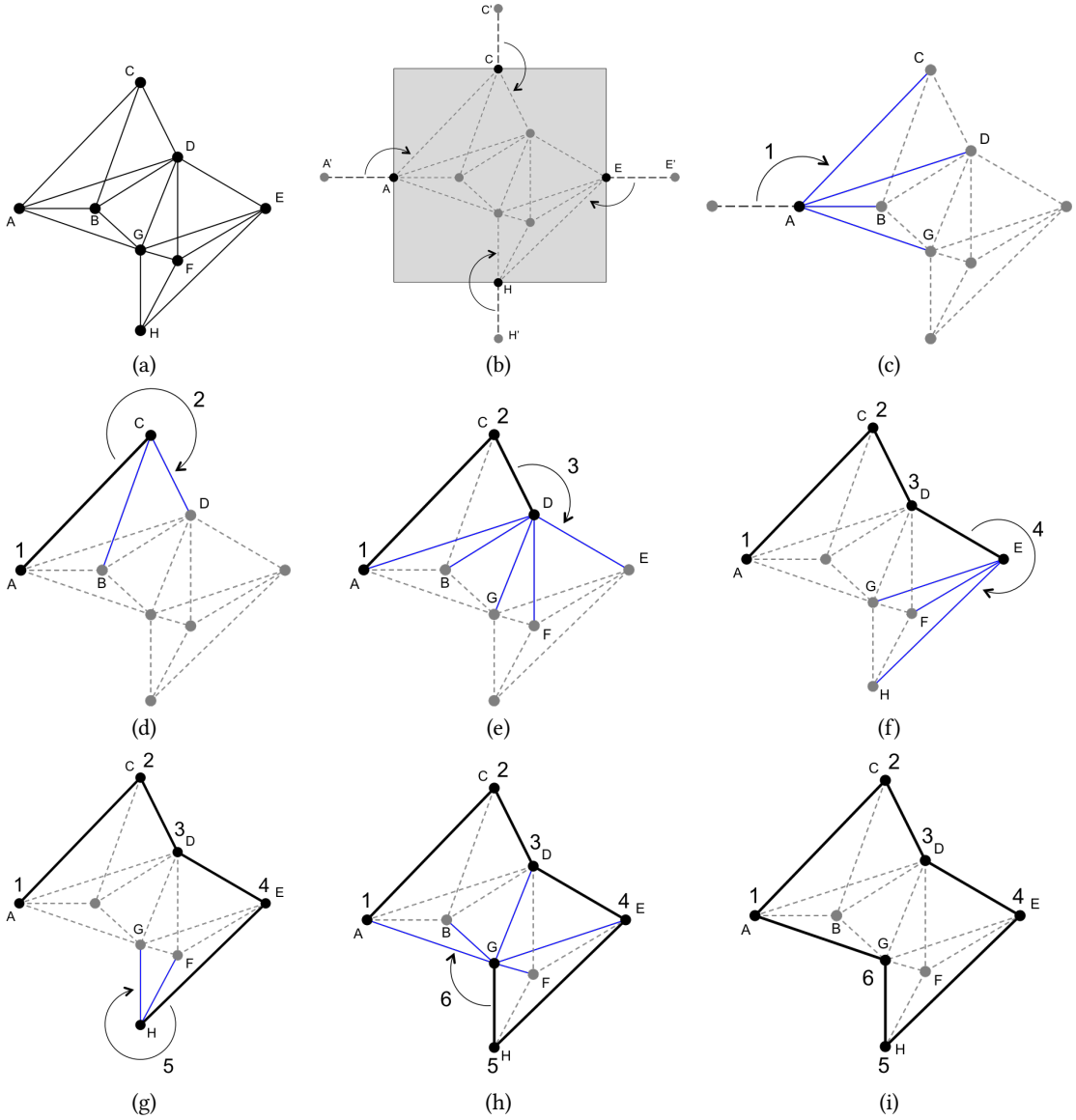

Figure 3: LPCN algorithm illustration.

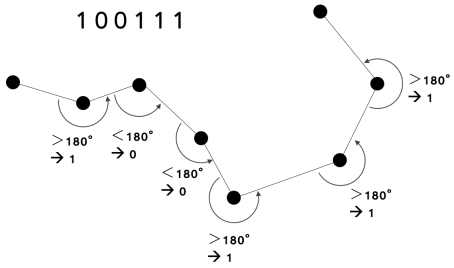

Figure 4: Generate binaries from angles.

5 EVALUATION

To measure the quality of the generated sequences of bits, we use the concept of entropy, first introduced by Shannon [20]. Sequences with higher entropy are considered to be of higher quality in terms of randomness and unpredictability.

The entropy of a generated sequence can be calculated using the formula:

$$H(X) = -\sum_{i=1}^{n} P(x_i) \cdot \log_2(P(x_i)) \quad (1)$$

Where:

$H(X)$: Entropy of the sequence
n : Number of possible bit values
x_i : Each possible bit value
$P(x_i)$: Probability of each bit value

We have generated many graphs with different connection ranges: 60, 100, 140, 180. For each graph, we have executed the LPCN algorithm with (version 1) and without (veresion 2) intersections to generate the binary sequence. Then, we have calculated the corresponging entropy.

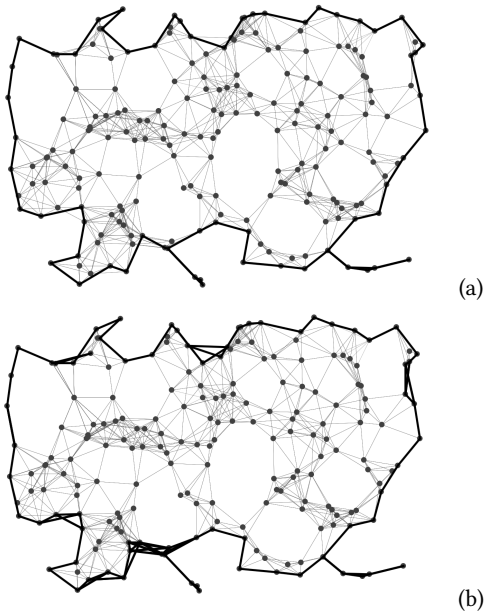

Figure 5: LPCN algorithm with (a) and without (b) intersections executed on a graph with 60 nodes.

Table 1 presents the computed entropy for binary sequences (numbers) generated from each graph. Instances with intersection (Yes) tend to exhibit slightly lower entropy values compared to those without intersection (No), suggesting a potential influence on the randomness of the sequences. Additionally, the variation in entropy across different graph sizes indicates the sensitivity of entropy to graph characteristics, highlighting the importance of considering both intersection and graph size when assessing the quality of the generated random sequences. In this experiment, the best entropy is obtained by the graphs with 60 and 100 nodes.

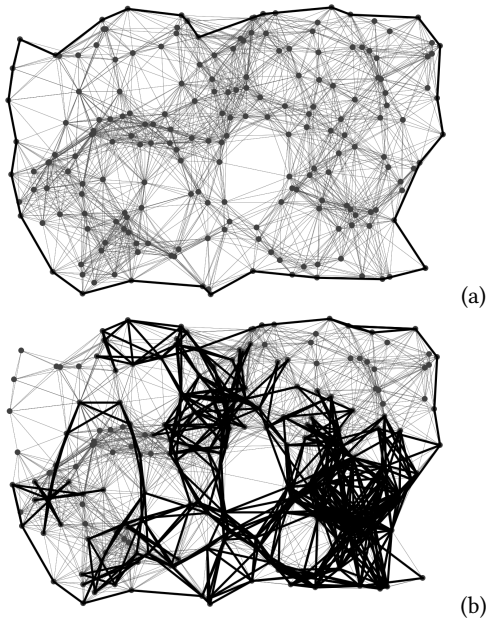

Figure 6: LPCN algorithm with (a) and without (b) intersections executed on a graph with 100 nodes.

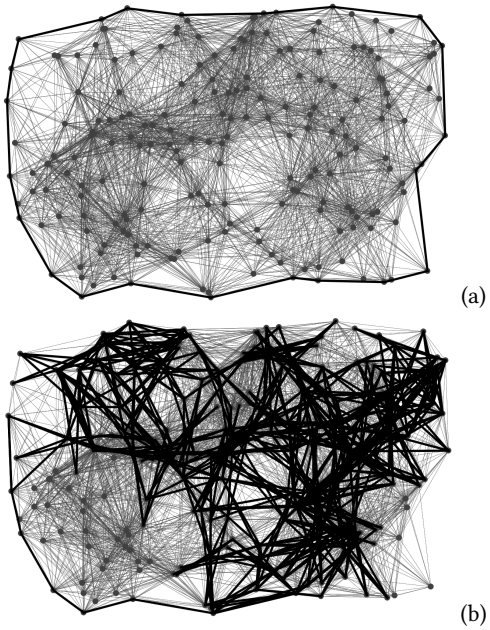

Figure 7: LPCN algorithm with (a) and without (b) intersections executed on a graph with 140 nodes.

6 CONCLUSIONS AND FUTURE WORK

The LPCN algorithm offers a promising approach to generating secure binary sequences for cryptographic purposes. By leveraging the unique node selection based on polar angles, this method enhances the unpredictability and security of generated sequences.

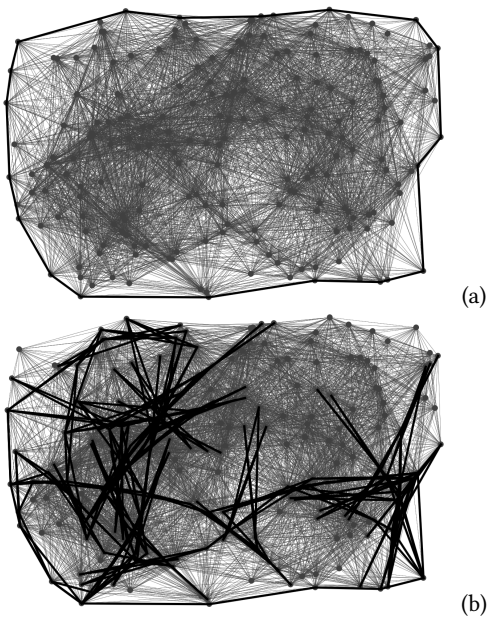

Figure 8: LPCN algorithm with (a) and without (b) intersections executed on a graph with 180 nodes.

Table 1: Computed Entropy

Inter.	Gr Size	Number Size	Entropy (bits)
Yes	60	57	2.7733
No	60	102	2.8810
Yes	100	33	2.8621
No	100	1026	2.7226
Yes	140	33	2.3540
No	140	1131	2.5207
Yes	180	33	1.8437
No	140	180	2.6371

Furthermore, the ability to transform graph images into concealed binary codes opens up innovative possibilities for data protection and secure communication. Future research in this area may focus on optimizing the LPCN algorithm for specific cryptographic applications, studiying the structure of graphs generating best binary numbers and exploring additional methods for binary sequence extraction from graphical representations.

REFERENCES

[1] Yas Abbas Alsultanny. 2008. Random-bit sequence generation from image data. *Image Vis. Comput.* 26, 4 (2008), 592–601. https://doi.org/10.1016/j.imavis.2007.07.008

[2] Emna Amri, Yacine Felk, Damien Stucki, Jiaju Ma, and Eric R. Fossum. 2016. Quantum Random Number Generation Using a Quanta Image Sensor. *Sensors* 16, 7 (2016), 1002. https://doi.org/10.3390/s16071002

[3] Rajarathnam Chandramouli and Nasir D. Memon. 2001. Analysis of LSB based image steganography techniques. In *Proceedings of the 2001 International Conference on Image Processing, ICIP 2001, Thessaloniki, Greece, October 7-10, 2001*. IEEE, 1019–1022. https://doi.org/10.1109/ICIP.2001.958299

[4] Abbas Cheddad, Joan Condell, Kevin Curran, and Paul McKevitt. 2010. Digital image steganography: Survey and analysis of current methods. *Signal Process.* 90, 3 (2010), 727–752. https://doi.org/10.1016/j.sigpro.2009.08.010

[5] Jessica Fridrich. 2009. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press. https://doi.org/10.1017/CBO9781139192903

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM* 63, 11 (2020), 139–144. https://doi.org/10.1145/3422622

[7] K. Gurunathan and S. P. Rajagopalan. 2020. A stegano - visual cryptography technique for multimedia security. *Multim. Tools Appl.* 79, 5-6 (2020), 3893–3911. https://doi.org/10.1007/s11042-019-7471-1

[8] Jiankun Hu and Fengling Han. 2009. A pixel-based scrambling scheme for digital medical images protection. *J. Netw. Comput. Appl.* 32, 4 (2009), 788–794. https://doi.org/10.1016/J.JNCA.2009.02.009

[9] Dyala R. Ibrahim, Je Sen Teh, and Rosni Abdullah. 2021. An overview of visual cryptography techniques. *Multim. Tools Appl.* 80, 21 (2021), 31927–31952. https://doi.org/10.1007/S11042-021-11229-9

[10] Inas Jawad Kadhim, Prashan Premaratne, Peter James Vial, and Brendan Halloran. 2019. Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research. *Neurocomputing* 335 (2019), 299–326. https://doi.org/10.1016/j.neucom.2018.06.075

[11] Mostefa Kara, Abdelkader Laouid, Muath AlShaikh, Mohammad Hammoudeh, Ahcene Bounceur, Reinhardt Euler, Abdelfattah Amamra, and Brahim Laouid. 2021. A compute and wait in pow (cw-pow) consensus algorithm for preserving energy consumption. *Applied Sciences* 11, 15 (2021), 6750.

[12] Takuya Kawashima and Shinichiro Mito. 2020. Random number generation using magnetic domain images of magneto-optical materials. *Japanese Journal of Applied Physics* 59, SE (2020), SEEA07.

[13] Apoorva Lakshman, Parkala Vishnu Bharadwaj Bayari, and Gaurav Bhatnagar. 2022. A novel logo-inspired chaotic random number generator. In *Proceedings of Academia-Industry Consortium for Data Science: AICDS 2020*. Springer, 297–306.

[14] Farid Lalem, Ahcène Bounceur, Madani Bezoui, Massinissa Saoudi, Reinhardt Euler, Tahar Kechadi, and Marc Sevaux. 2017. LPCN: Least polar-angle connected node algorithm to find a polygon hull in a connected euclidean graph. *Journal of Network and Computer Applications* 93 (2017), 38–50. https://doi.org/10.1016/j.jnca.2017.05.005

[15] Dragan Lambić and Mladen Nikolić. 2017. Pseudo-random number generator based on discrete-space chaotic map. *Nonlinear Dynamics* 90, 1 (2017), 223–232.

[16] Rongzhong Li. 2015. A True Random Number Generator algorithm from digital camera image noise for varying lighting conditions. In *SoutheastCon 2015*. 1–8. https://doi.org/10.1109/SECON.2015.7132901

[17] Jia Liu, Yan Ke, Zhuo Zhang, Yu Lei, Jun Li, Minqing Zhang, and Xiaoyuan Yang. 2020. Recent Advances of Image Steganography With Generative Adversarial Networks. *IEEE Access* 8 (2020), 60575–60597. https://doi.org/10.1109/ACCESS.2020.2983175

[18] Congduc Pham, Ahcène Bounceur, Laurent Clavier, Umber Noreen, and Muhammad Ehsan. 2020. 4 - Radio channel access challenges in LoRa low-power wide-area networks. In *LPWAN Technologies for IoT and M2M Applications*, Bharat S. Chaudhari and Marco Zennaro (Eds.). Academic Press, 65–102. https://doi.org/10.1016/B978-0-12-818880-4.00004-1

[19] Massinissa Saoudi, Farid Lalem, Ahcène Bounceur, Reinhardt Euler, M-Tahar Kechadi, Abdelkader Laouid, Madani Bezoui, and Marc Sevaux. 2017. D-LPCN: A distributed least polar-angle connected node algorithm for finding the boundary of a wireless sensor network. *Ad Hoc Networks* 56 (2017), 56–71. https://doi.org/10.1016/j.adhoc.2016.11.010

[20] Claude E. Shannon. 1948. A mathematical theory of communication. *Bell Syst. Tech. J.* 27, 3 (1948), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

[21] Yanting Wang, Mingwei Tang, and Zhen Wang. 2020. High-capacity adaptive steganography based on LSB and Hamming code. *Optik* 213 (2020), 164685. https://doi.org/10.1016/j.ijleo.2020.164685

[22] Qing Zhou, Xiaofeng Liao, Kwok-Wo Wong, Yue Hu, and Di Xiao. 2009. True random number generator based on mouse movement and chaotic hash function. *Inf. Sci.* 179, 19 (2009), 3442–3450. https://doi.org/10.1016/J.INS.2009.06.005