International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

A Novel Corona Graph Based Proof-of-Work Algorithm for Public Blockchains

Shalini Agarwal

Submitted: 24/12/2023 **Revised**: 30/01/2024 **Accepted**: 06/02/2024

Abstract—Various mathematical puzzle-based Proof-of-Work (PoW) algorithms have been developed to prevent adversaries from adding illegitimate nodes in public blockchains. The puzzles are devised in such a manner that they are difficult to solve by the blockchain members, hereto called 'provers', but easily verifiable for correctness by the blockchain administrator, hereto called as 'verifier'. However, existing algorithms are either very computationally intensive requiring high end processing capability or consume a lot of space for storing homogeneous data tables at prover and verifier ends. In this article, we introduce a novel Corona Graph based PoW technique for distinguishing adversaries from legitimate blockchain clients and hence mitigating addition of illegitimate nodes in public blockchains. The proposed algorithm exploits the properties of modular inverses of integers in such a way so as to create a tradeoff between computational complexity and storage space required. The values of the parameters can be adjusted to set the difficulty level of corona graph according to application requirements ranging from highly critical to common client server applications. We provide formal proof of correctness of our approach and demonstrate analytically that the proposed technique is resilient and offers a practical solution for identifying adversaries.

Keywords and Phrases: *Blockchain, Consensus, Corona Graphs, Cryptographic Puzzles, Proof-of-Work*

1 Introduction

The backbone of any blockchain model is a decentralized ledger that records each and every transaction among the participants of transactions [1]. When a new transaction takes place, a new 'block' with transaction details is generated and added to the 'chain' and hence the name 'Blockchain' originates. The task of validating a new block requires a lot of computational effort and is accomplished by miners. The miners compete with each other to validate the next block and win a reward by solving a cryptographic puzzle and producing a Proof-of-work (PoW). In cryptocurrency mining, specialized machines such as ASICs are utilized to compute solutions of complex and secure hash algorithms, for example, SHA-256. Although these mechanisms are very strong but they violate fairness of the system as they prevent commodity miners from participating in the block validation process [2]. This problem has inspired researchers to study alternative models of PoW which allows decentralized nodes with constrained hardware resources, to compete for a shared resource over the Internet.

Cryptographic puzzles were originally proposed to combat spams [3]. Later they were improvised to defend against Denial of Service (DoS) attacks and Sybil attacks. Researchers proposed different ways of constructing and distributing puzzles in which a client is subjected to solve a moderately hard computational problem and present the solution to the server as a proof of work before the server grants any significant amount of resource on the client's request [4][5]. Solving a puzzle involves performing significant iterations of cryptographic operations, such as hashing or modular arithmetic. Consequently, the more a client demands from the server, the more difficult puzzle it has to solve, thus expending its own computational resources. Puzzles are so designed so that their construction and verification at the server end does not increase the server load. In other words, puzzles are difficult to solve by the client or 'prover' but easy to verify by the server or 'verifier'. Literature outlines eight essential attributes for an easy to verify cryptographic puzzle. They are- Computation guarantee, Adjustability of difficulty, Correlation-free, Stateless, Tamper-resistance, Non-

Amity University, Uttar Pradesh,
drsagarwal.04@gmail.com

parallelizability, Puzzle Fairness and Minimum Interference [3]. Graphs based puzzles are quite interesting and fulfill the aforementioned attributes. The recent past has seen a growing interest in exploring graphs as a tool to propose innovative methodologies in the domain of security and cryptography [6][7]. This paper proposes a novel algorithm as an effective alternative implementation of proof-of-work in public blockchains, using special corona graphs and certain algebraic computations. The proposed algorithm leads to a computationally intensive puzzle which is difficult to solve but easy to verify. Moreover, the values of the graph vertices can be adjusted and henceforth the algorithm may be used for security critical blockchain application as well as day to day client server applications. The steps are simple to implement but guarantee that legitimate users are identified from the adversaries. Hence this algorithm has a wide scope for implementation in a variety of distributed system setups

The rest of the paper is organized as follows. Section 2 summarizes related work. In section 3 the Corona Graph based proof-of-work algorithm along with numerical example formal proof is illustrated. Section 4 presents conclusion and future work.

2 Related Work

In the seminal work [3] the concept of computing a moderately hard function by the client was first introduced. However, this mechanism is not suitable for secure PoW algorithm due to its vulnerability to solution pre-computations. As a counter measure to this mechanism, a hash function-based puzzle scheme called client puzzles has been proposed by [8]. Client puzzles effectively resolves the problem of puzzle pre-computation. In [9] a different implementation of puzzles is suggested in the form of puzzle auction that defends denial-of-service attacks against authentication protocols and prevention of TCP SYN flooding. The authors highlighted the weaknesses of hard functions and improvised solution for the problem by including small contribution from the client side during puzzle generation. Another promising work by [10] proposed outsourcing of puzzle distribution to external agents for enhanced security. The authors in [11] suggested a method similar to the puzzle auction to cater denial-of-capability attacks which prevent clients from setting up capabilities and sending prioritized packets in the network. Effectively, clients who volunteer to solve harder puzzles requiring complex computations are given higher priority. In all the aforementioned proposals above, finding the correct puzzle solution is parallelizable. Therefore, an adversary may obtain the solution faster by computing it in parallel using multiple powerful machines. In addition, most of them suffer from resource disparity problem and interferes with the concurrently running user applications.

Graph based puzzles as proof-of-work are non-parallelizable and do not have the problems of resource disparity and interference with user applications. In fact, interaction between graph theory and proof-of-work algorithms is quite interesting. Applications of graph theory in security and blockchain has been extensively studied in [12] in which the authors present an innovative algorithm for encryption and decryption using connected graphs. In [13], the authors have proposed a graph-based encryption algorithm in which fundamental circuits are chosen with respect to corresponding weights of edges. In [14][15], authors present unique method of transferring data by using bipartite graph. Furthermore, they also suggested novel bipartite graph-based propagation approach to overcome fraud detection in large advertising system. In similar works, security models using cycle graph, complete graph, and minimum spanning tree have been developed. The methodology given in [16][18], used coset diagram on projective line over the finite field F to construct proposed substitution box (S-box). The mechanism generates a strong S-box using orbits of coset graphs and the action of the symmetric group thus presenting a very efficient algorithm for encrypting a graph for its secure transmission from a sender to a receiver.

The literature study reflects that PoW algorithms based on Corona graphs have not been explored by researchers till date [17]. Corona graph based PoW algorithm offer a resilient approach that can be used in blockchains to achieve consensus and secure the communication in the network. This paper aims to describe a novel PoW algorithm based on computations on corona graph $C_n \odot K_1$. The proposed algorithm requires the prover to compute a sequence of modular inverses on a specific corona graph. The verifier verifies the computed inverses for its correctness and eventually allows the prover

to add new block to the blockchain. The mechanism requires blockchain administrators to construct a graph based on corona operations on encrypted texts.

The modular arithmetic involved, demands intensive computational power which the blockchain users must expend in order to solve the graph. Users compete to solve the graph and the one who finds the solution first, gets the chance to add a new block to the blockchain and is rewarded. It is highly secure due to its computational requirements. To perform a successful attack, an attacker would need to control over 51% of the network's computational power. This mechanism also promotes decentralization as anyone with sufficient computational resources can participate in the process.

3 Novel Corona Graph Based Proof Of Work Algorithm

The corona of two graphs G and H is denoted by G⊙ H, which is formed by taking one copy of G and |V(G)| copies of H and joining the i^{th} vertex of G to each vertex in the i^{th} copy of H. This product is an important operation between graphs, and was introduced by Frucht and Harary. The corona graph of the cycle C_n with K_1, written as, $C_n \odot K_1$, is a graph consisting of 2n vertices obtained by attaching n pendant edges in a cycle graph C_n. The algorithm starts with taking a plaintext which will be used as a key in the corona graph computation. Every letter a_i in the predetermined text shall be encoded numerically according to the encoding table given as Table 1.

Table 1: Letter Encoding Table

Letter	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Encoding	1	2	3	4	5	6	7	8	9	10	11	12	13
Letter	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Encoding	14	15	16	17	18	19	20	21	22	23	24	24	26

The next step is to transmute the numeric sequence up to n-place, through shift cipher resulting into new numeric values b_i. By randomly selecting some positive integers c_i which are relatively prime to b_i, construct corona graph $C_n \odot K_1$ and allot value of c_i to main vertices randomly. Now compute modular inverses d_i as $d_i=(b_i)^{-1}\mod(c_i)$, and attach d_i to suspended pendant vertices of $C_n \odot K_1$ as shown in Figure 1. The resulting corona graph is sent to the prover for computing back original text and eventually winning the chance to add a node to the blockchain.

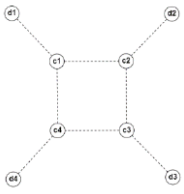

Fig 1: Structure of Corona Graph

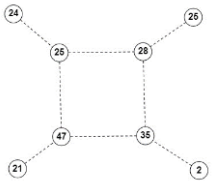

Fig 2: Sample Corona graph for the text 'GENE'

3.1 Corona Graph Construction Algorithm

Steps to construct Corona Graph are summarized in Algorithm 1. The blockchain administrator constructs the Corona graph with 2n vertices and sends it to all the users of the blockchain. All users must solve this graph to find the original text and return it to the administrator for verification. In literature, administrator is also called as a verifier.

Creating n copies of graph K_1 involves iterating over each vertex in C_n and then connecting each

vertex in C_n to K_1. Thus, time complexity of Corona Graph construction algorithm is $O(n^2)$.

Algorithm 1: Corona Graph Construction for given text

Input: Plaintext T of length n

Assign each letter a_i of T, a numerical value b_i according encoding table

for each a_i in T **do**

  $b_i = a_i + n \pmod{26}$

end

Find a sequence of positive integers $c_1, c_2, \ldots, c_n$ ($c_1 < c_2 < \ldots < c_n$ and $c_i > 26$), such that $GCD(c_i, b_i)=1$

Construct Corona Graph C with 2n vertices

for each vertex non pendent vertex i in C **do**

  Assign weight c_i to vertex adjacent to i^{th} pendent vertex in C

end

for each pendent vertex i in C **do**

  compute $d_i=(b_i)^{-1} \bmod (c_i)$

  Assign weight d_i to i^{th} pendent vertex in C

end

return C

3.2 Text Reconstruction Algorithm

Steps To Reconstruct Plaintext From Corona Graph Are Summarized In Algorithm 2. The Blockchain User Reconstructs The Plaintext W Sends It To The Blockchain Administrator. If Original Text T And Reconstructed Text W Match, The Administrator Allows The User To Add A New Block To The Chain. Users Or Graph Solvers Must Return The Reconstructed Text To The Verifier Within A Stipulated Time Interval. The One Who Does So In The Shortest Time Wins A Chance To Add Block To The Chain.

The Time Complexity Of Computing The Modular Inverse (Inverse Modulo) Depends On The Algorithm Used. One Common And Efficient Algorithm For Finding The Modular Inverse Is The Extended Euclidean Algorithm. The Extended Euclidean Algorithm Has A Time Complexity Of Log (Min (A, B))) Making It Efficient For Large Integers. The Algorithm Involves Repeated Application Of The Euclidean Algorithm Along With Backward Substitutions To Find The Bézout Coefficients.

Algorithm 2: Text Reconstruction from Corona Graph

Input: Corona Graph C

for each vertex non pendent vertex i in C **do**

Sort $c_1, c_2, \ldots c_n$ in increasing order

end

for each pendent vertex i in C **do**

compute inverses $b_i = (d_i)^{-1} \text{mod}(c_i)$

compute $w_i = b_i - (2n/2) \bmod 26$

end

Reconstruct plaintext w

If w=T

User is permitted to added a block to the chain

else

User is an attacker

endif

3.3 Numerical example for Corona Graph Construction and Text Reconstruction and performance analysis

Table 2: Letter Encoding of text 'GENE'

Letter	a_i	**Numeric Encoding**	**shift cipher** $b_i = a_i + n \pmod{26}$	b_i	**New Text**
G	a_1	7	7+4=11	b_1	K
E	a_2	5	5+4=9	b_2	I
N	a_3	14	14+4=18	b_3	R
E	a_4	5	5+4=9	b_4	I

Assume that the pre-determined plaintext T is 'GENE' of length n=4. Applying numeric encoding followed by shift cipher $b_i = a_i + n \pmod{26}$, on each letter a_i we get new numeric sequences b_i as shown in Table 2. Now, to construct corona graph for the encrypted text we select random increasing integers c_i ($c_i > 26$) such that GCD (c_i, b_i) =1. We select c_1=25, c_2=28, c_3=35 and c_4=47 since, GCD (25,11) = GCD (28,9) = GCD (35,18) = GCD (47,9) =1. We use these values to give weights to the non-pendent vertices of the corona graph. For computing weights of the pendent vertices of the corona graph, we again perform inverse modular function $d_i=(b_i)^{-1} \text{mod}(c_i)$. Thus, we obtain d_1=16, d_2=25, d_3=2 and d_4=21. The corona graph is now constructed with the obtained values of c_i and d_i as shown in Figure 2. This labeled graph (Figure 2) is sent to the blockchain users. The recipients, after receiving that labeled graph, arrange the main vertices in ascending order i.e. 25<28 < 35 < 47. Taking inverses of pendent vertices with respect to the corresponding values of c_i, cipher text is obtained as shown in equations 1(a) to 1(d) and further reconstructed using eq. 2(a) to 2(d).

$$11 = (16)^{-1} \bmod (25) \ldots\ldots 1 \text{ (a)}$$

$$9 = (25)^{-1} \bmod (28) \ldots\ldots 1 \text{ (b)}$$

$$18 = (2)^{-1} \bmod (35) \ldots\ldots 1 \text{ (c)}$$

$$9 = (21)^{-1} \bmod (47) \ldots\ldots 1 \text{ (d)}$$

$w_1 = 11 - (2*4/2) \bmod 26 = 7$ (Letter G) ……. 2 (a)

$w_2 = 9 - (2*4/2) \bmod 26 = 5$ (Letter E) …….2 (b)

$w_3 = 18 - (2*4/2) \bmod 26 =14$ (Letter N) ……..…. 2 (c)

$w_4 = 9 – (2*4/2) \bmod 26 = 5$ (Letter E)

……2 (d)

4 Conclusion And Future Work

This work explored application of Corona Graph in implementing effective proof-of-work algorithm that can be used securely in public blockchain as an alternative to the conventional SHA algorithm which is of rather higher complexity. Since, the time complexity of the corona operation to depends on the size of the input graphs involved, the proposed algorithm can be suitably scaled-up to increase the computational effort, a prover must put in to reconstruct the original text from the Corona Graph. Users return a solution to the corona graph and verifiers collect all the solutions which arrive within a stipulated block interval time. Verifier selects the solution with the lowest response time and gives permission to the solver to add a new block to the chain.

This work considered corona operation between cycle graph C_n and complete graph K_1. In future, corona operations between more complex graphs will be studied to enhance hardness of the problem which will be more difficult for the solver to solve yet easy to verify by the verifier.

References

[1] Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System." Available at: https://bitcoin.org/bitcoin.pdf

[2] Mehmud Abliz, Taieb Znati and Adam J. Lee, “Mitigating Distributed Service Flooding Attacks with Guided Tour Puzzles, International Journal on Advances in Security’, vol 5 no 3 & 4, year 2012, http://www.iariajournals.org/security/

[3] Dwork, C., & Naor, M. (1993). Pricing via Processing or Combatting Junk Mail. Advances in Cryptology - CRYPTO’92: Lecture Notes in Computer Science, 740, 139-147.

[4] Gervais, A., Ritzdorf, H., Karame, G. O., & Capkun, S. (2015). Tampering with the Delivery of Blocks and Transactions in Bitcoin. CCS ’15 Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (pp. 692-705). Denver, Colorado, USA: ACM.

[5] Meshkov, D., Chepurnoy, A., & Jansen, M. (2017). Short Paper: Revisiting Difficulty Control for Blockchain Systems. In Data Privacy Management, Cryptocurrencies and Blockchain Technology (pp. 429-436). Oslo, Norway

[6] W. Mahmoud and A. Etaiwi, “Encryption algorithm using graph theory,” Journal of Scientific Research and Reports, vol. 3, no. 19, pp. 2519–2527, 2014.

[7] Werner, F. Graph-Theoretic Problems and Their New Applications. Mathematics 2020, 8, 445. https://doi.org/10.3390/math8030445

[8] A. Juels and J. Brainard, “Client puzzles: A cryptographic countermeasure against connection depletion attacks,” in NDSS ’99, San Diego, CA, 1999, pp. 151–165

[9] X. Wang and M. K. Reiter, “Defending against denial-of service attacks with puzzle auctions,” in IEEE Symposium on Security and Privacy, Oakland, 2003, pp. 78–92

[10] Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S., & Saxena, P. (2016). "A Secure Sharding Protocol for Open Blockchains." arXiv preprint arXiv:1704.08502.

[11] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y. Hu, “Portcullis: Protecting connection setup from denial-of-capability attacks,” in ACM SIGCOMM, 2007, pp. 289– 300

[12] G. A. Selim, “How to encrypt a graph,” International Journal of Parallel, Emergent and Distributed Systems, vol. 35, no. 6, pp. 668–681, 2020.

[13] P. Kedia and S. Agrawal, “Encryption using Venn-diagrams and graph,” International Journal of Advanced Computer Technology, vol. 4, no. 01, pp. 94–99, 2015

[14] M. Yamuna and K. Karthika, “Data transfer using bipartite graphs,” International Journal of Advance Research in Science and Engineering, vol. 4, no. 02, pp. 128–131, 2015.

[15] B. R. Arunkumar, “Applications of Bipartite Graph in diverse fields including cloud computing,” International Journal of Modern Engineering Research, vol. 5, no. 7, p. 7, 2015

[16] A. Razaq, H. Alolaiyan, M. Ahmad et al., “A novel method for generation of strong substitution-boxes based on coset graphs and symmetric groups,” IEEE Access, vol. 8, pp. 75473–75490, 2020.

[17] R. Frucht and F. Harary, “On the corona of two graphs,” Aequationes Math, vol. 4, pp. 322–325, 1970.Garay, J. A., Kiayias, A., & Leonardos, N. (2015). "The Bitcoin Backbone Protocol: Analysis and Applications."